VŠB – Technická univerzita Ostrava
Univerzitní studijní programy
Katedra matematiky a deskriptivní geometrie

Sítě pro metodu konečných prvků a generování matic tuhosti

Finite element meshes and assembling of stiffness matrices

Vypracoval: Vladimír Arzt
Vedoucí práce: Mgr. Zuzana Morávková, Ph.D.

Ostrava 2019

VŠB - Technická univerzita Ostrava
Univerzitní studijní programy
Katedra matematiky a deskriptivní geometrie

Zadání bakalářské práce

Student: **Vladimír Arzt**

Studijní program: B3968 Aplikované vědy a technologie

Studijní obor: 3901R076 Aplikované vědy a technologie

Téma: Sítě pro metodu konečných prvků a generování matic tuhosti
Finite element meshes and assembling of stiffness matrices

Jazyk vypracování: čeština

Zásady pro vypracování:

Cílem bakalářské práce je seznámit se s vytvářením sítí a matic tuhosti při řešení diferenciálních rovnic metodou konečných prvků. V současnosti existuje řada volně dostupných generátorů, které je potřeba se naučit používat. Další krokem je zvládnutí práce s již vygenerovanou sítí pomocí vhodných datových struktur, které umožní vytvořit co nejefektivnější a nejpřehlednější programové kódy pro sestavování matic tuhosti. Některé kódy lze vektorizovat. Výsledkem bakalářské práce bude zvládnutí „preprocessingu" metody konečných prvků. Samotné řešení diskrétních algebraických úloh bude mít okrajový význam jako potvrzení správnosti sestavených objektů.

Seznam doporučené odborné literatury:

[1] MESH2D - Delaunay-based unstructured mesh-generation, v Matlabu, https://www.mathworks.com/matlabcentral/fileexchange/25555-mesh2d-delaunay-based-unstructured-mesh-generation.
[2] ISO2MESH, volný toolbox pro Matlab, http://iso2mesh.sourceforge.net/cgi-bin/index.cgi.
[3] R. Blaheta: Matematické modelování a metoda konečných prvků, http://mi21.vsb.cz/sites/mi21.vsb.cz/files/unit/numericke_metody_2.pdf.
[4] J. Koko: Vectorized Matlab Codes for the Stokes Problem with P1-Bubble/P1 Finite Element, https://www.isima.fr/~jkoko/Codes/StokesP1BubbleP1.pdf.

Formální náležitosti a rozsah bakalářské práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí bakalářské práce: **Mgr. Zuzana Morávková, Ph.D.**

Datum zadání: 08.12.2017
Datum odevzdání: 20.05.2019

VYSOKÁ ŠKOLA BÁŇSKÁ - TECHNICKÁ UNIVERZITA OSTRAVA UNIVERZITNÍ STUDIJNÍ PROGRAMY

vedoucí katedry

Ing. Zdeňka Chmelíková, Ph.D.
prorektorka pro studium

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

V Ostravě 20. května 2019

Vladimír Arzt

Rád bych poděkoval vedoucí této bakalářské práce Mgr. Zuzaně Morávkové, Ph.D. za nesmírnou ochotu a trpělivost při jejím vzniku. Dále prof. RNDr. Radku Kučerovi, Ph.D za pečlivé pročtení textu a za cenné připomínky a komentáře. Také bych chtěl poděkovat své rodině, která to se mnou měla po nesčetných dlouhých studijních nocích v průběhu studia těžké.

Abstrakt

Bakalářská práce se zabývá tvorbou sítí pro metodu konečných prvků a generováním matic tuhosti. V rámci řešení bakalářské práce byla provedena řada testů v 1D, 2D a 3D úlohách pro určení výpočetní náročnost generování matic tuhosti standardním a vektorizovaným způsobem. První část je zaměřena na jednodimenzionální úlohu, kde se sít vytváří pouhým rozdělením intervalu na subintervaly. Druhá část se věnuje dvoudimenzionální úloze kde je sít vytvářena balíkem MESH2D, navíc je zde aproximační množina rozšířená o bublinkové funkce. Poslední část je zaměřena na třídimenzionální úlohu. Sít je vytvořena balíkem ISO2MESH a aproximační množina je také obohacena o bublinkové funkce.

Klíčová slova: Metoda konečných prvků, matice tuhosti, MESH2D, ISO2MESH, bublinková funkce

Abstract

The bachelor thesis deals with meshing for finite element method and stiffness matrix generation. The series of tests in 1D, 2D and 3D tasks are performed to determine the compute complexity of standart and vectorized generating stiffness matrices. The first part is focused on a onedimensional task, where the mesh is created by simply dividing the interval into subintervals. The second part deals with a twodimensional problem where the mesh is created by the MESH2D package. In addition, the approximation set is extended by bubble functions. The last part is focused on a threedimensional task. Mesh is created by the ISO2MESH package and the approximation set is also enriched with bubble functions.

Key Words: Finite Flement Method, stiffness matrix, MESH2D, ISO2MESH, bubble function

Obsah

Seznam použitých funkcí, prostorů a symbolů

Ω	–	Oblast Ω
$\partial\Omega$	–	Hranice oblasti Ω
$\bar{\Omega}$	–	Uzávěr oblasti Ω
$H^1(\Omega)$	–	Sobolevův prostor funkcí
$H_0^1(\Omega)$	–	Sobolevův prostor funkcí, jejichž stopa je na hranici $\partial\Omega$ nulová
$C(\bar{\Omega})$	–	Prostor funkcí, jenž jsou spojitě prodloužitelné až do hranice $\partial\Omega$
$L^2(\bar{\Omega})$	–	Prostor lebesgueovsky integrovatelných funkcí v druhé mocnině
$P_1(T)$	–	Polynomy nejvýše prvního stupně na prvku T
$\mathcal{T}, \mathcal{T}_h$	–	Triangulace
$v_{\|T}$	–	Restrikce funkce na prvek T
Δ	–	Laplaceův operátor $\Delta = \nabla^2 = \nabla \cdot \nabla =$ div grad
$\nabla \cdot \boldsymbol{u}$	–	Divergence
$\oplus$	–	Direktní součet

Seznam obrázků

Seznam tabulek

Seznam výpisů zdrojového kódu

1 Úvod

Metoda konečných prvků prošla od padesátých let 20. století značným vývojem, který byl způsoben rostoucími možnostmi uplatnění díky rychlému navyšování výkonu číslicových počítačů. Díky tomu se začalo MKP využívat ve strojírenství, stavebnictví, ekonomice, medicíně a v mnoha dalších odvětvích, což nadále podporovalo její vývoj.

Princip MKP spočívá v rozložení zadané oblasti na mnoho menších podoblastí, na kterých se pak počítají potřebné vlastnosti. Jelikož se jedná o numerickou metodu, tak je řešení zatíženo určitou odchylkou od přesného řešení. Tuto odchylku lze nejsnadněji snížit zvětšením hustoty sítě, tedy zvýšením počtu prvků. To je podmíněno nárůstem výpočetních a paměťových nároků pro získání řešení.

V této práci se budeme zabývat „preprocessingem“ metody konečných prvků, ve kterém dochází ke generování výpočetní sítě, matic a vektoru pravé strany. Následné řešení takto sestavené soustavy má jen okrajový význam pro ověření správnosti sestavených objektů.

Sestavení matic tuhosti lze provést standardně, tedy průchodem přes všechny prvky za pomocí cyklů. Rozložením standardního sestavení na jednotlivé cykly lze najít určité zákonitosti umožňující toto sestavení vektorizovat. Vektorizované sestavení díky absenci cyklů pak zabírá zlomek času.

Dále dochází během preprocessingu k aproximaci úlohy za pomocí lineárních bázových funkcí, přičemž tyto funkce mohou být rozšířeny o tzv. bublinkové funkce, které vychází z původních lineárních.

Popsána bude nejprve nejjednoduchší úloha v jedné dimenzi, ve které se pokusíme především o objasnění principu sestavení. V dalších kapitolách dojde k rozšíření do dvou a tří dimenzí, ve kterých mají úlohy stejný princip, ale s rostoucím počtem dimenzí jsou rozsáhlejší a umožňují tvorbu sítí na oblastech různých tvarů.

2 Modelová úloha 1D

V této kapitole si ukážeme na jednodimenzionální modelové úloze základní postupy sestavování soustav lineárních rovnic u metody konečných prvků, které lze provádět vektorizovaně. V dalších kapitolách pak tyto postupy zobecníme pro vícedimenzionální úlohy.

2.1 Formulace úlohy

Budeme uvažovat okrajovou úlohu pro obyčejnou diferenciální rovnici druhého řádu:

$$-u'' = f \text{ v } (a,b),\ u(a) = u_a,\ u(b) = u_b, \tag{1}$$

kde u je hledaná funkce, f je zadaná pravá strana a u_a, u_b jsou předepsané Dirichletovy okrajové podmínky. Klasickým řešením úlohy (1) nazýváme funkci u, která je dvakrát spojitě diferencovatelná na intervalu (a,b), vyhovuje dané diferenciální rovnici a její spojité prodloužení na hranici intervalu (a,b) splňuje Dirichletovy okrajové podmínky. Klasické řešení úlohy (1) existuje, je-li pravá strana f spojitou funkcí na $\langle a,b\rangle$.

Při odvozování slabé formulace úlohy (1) použijeme integraci per partes a následně zeslabíme požadavky na hladkost řešení, viz dodatek A.1. Slabá formulace úlohy (1) má tento tvar:

$$\left.\begin{array}{l} \text{Najdi } u \in H^1(a,b),\ u(a) = u_a,\ u(b) = u_b \text{ tak, že} \\ \displaystyle\int_a^b u'v' = \int_a^b fv \quad \forall v \in H_0^1(a,b). \end{array}\right\} \ (P)$$

Úlohu (P) budeme aproximovat metodou konečných prvků:

$$\left.\begin{array}{l} \text{Najdi } u_h \in V_h \text{ tak, že} \\ \displaystyle\int_a^b u_h'v_h' = \int_a^b fv_h \quad \forall v_h \in V_{0h}, \end{array}\right\} \ (P_h)$$

kde V_h je konečnědimenzionální aproximace prostoru $H^1(a,b)$ zahrnující okrajové podmínky a V_{0h} je konečnědimenzionální aproximace prostoru $H_0^1(a,b)$.

Aproximační prostory vytvoříme pomocí sítě uzlů $\mathcal{P} = \{x_i :\ i = 0,\ldots,\ n+1\}$, kdy pro jednoduchost budeme uvažovat uspořádání $a = x_0 < x_1 < \ldots < x_{n+1} = b$. Symboly $h_i = x_{i+1} - x_i$, $i = 0,1,\ldots,n$, pak označíme kroky sítě. Dále budeme používat dělení intervalu $\langle a,b\rangle$ na množinu subintervalů $\mathcal{T}_h = \{T_i = \langle x_i, x_{i+1}\rangle :\ i = 0,1,\ldots,n\}$.

V MATLABu budeme tyto struktury reprezentovat následovně:

$$\texttt{p} = [x_0, x_1, \ldots, x_{n+1}]';\ \texttt{np} = \texttt{length(p)};$$
$$\texttt{h} = \texttt{p(2 : end)} - \texttt{p(1 : end - 1)};$$
$$\texttt{t} = [\texttt{1 : np} - \texttt{1; 2 : np}]';\ \texttt{nt} = \texttt{size(t, 1)};$$

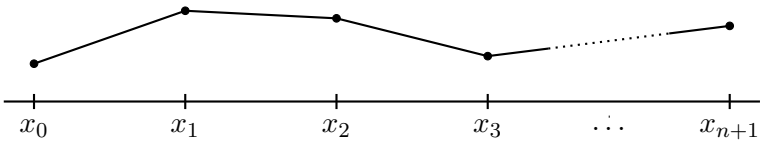
Subintervalům $T_i \in \mathcal{T}_h$ spolu s příslušnými bázovými funkcemi říkáme „konečné prvky“. Všimněme si, že dělení $\mathcal{T}_h$ reprezentujeme pouze indexovým polem `t`, které vytváří vazbu na uzly uložené v `p`.

2.2 Lineární konečné prvky (P1)

V tomto případě definujeme prostor V_h takto:

$$V_h = \{v_h \in C(\langle a, b\rangle) : \ v_{h|T} \in P_1(T)\ \forall T \in \mathcal{T}_h,\ v_h(a) = u_a,\ v_h(b) = v_b\}.$$

Grafem každé funkce $v_h \in V_h$ je lomená čára, viz obrázek 1.



Obrázek 1: Lomená čára

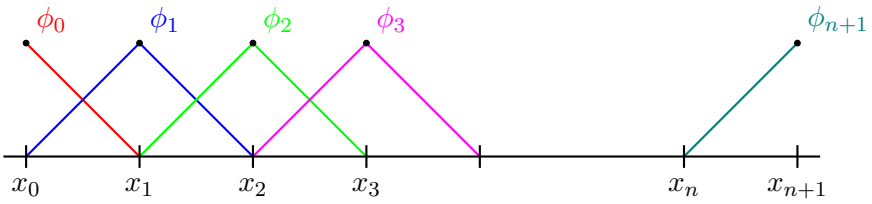
Funkci $v_h \in V_h$ můžeme reprezentovat pomocí jejich uzlových hodnot $v_i = v_h(x_i),\ i = 0, 1, \ldots, n+1$. Použijeme přitom po částech lineární bázové funkce $\phi_i,\ i = 0, 1, \ldots, n+1$ definované takto, viz obrázek 2,

$$\phi_0(x) = \begin{cases} h_0^{-1}(x_1 - x), & x \in \langle x_0, x_1\rangle, \\ 0, & \text{jinak;} \end{cases}$$

pro $i = 1, \ldots, n$

$$\phi_i(x) = \begin{cases} h_{i-1}^{-1}(x - x_{i-1}), & x \in \langle x_{i-1}, x_i\rangle, \\ h_i^{-1}(x_{i+1} - x), & x \in \langle x_i, x_{i+1}\rangle, \\ 0, & \text{jinak;} \end{cases}$$

$$\phi_{n+1}(x) = \begin{cases} h_n^{-1}(x - x_n), & x \in \langle x_n, x_{n+1}\rangle, \\ 0, & \text{jinak.} \end{cases}$$



Obrázek 2: Po částech lineární bázové funkce

Všimněme si, že každá bázová funkce ϕ_i je nenulová nejvýše na dvou sousedních intervalech z $\mathcal{T}_h$. Na každém intervalu $T_i \in \mathcal{T}_h$ jsou tedy nenulové právě dvě bázové funkce ϕ_i a ϕ_{i+1}, pro jejichž derivace uvnitř tohoto intervalu platí:

$$\phi_i'(x) = -h_i^{-1},\ \phi_{i+1}'(x) = h_i^{-1}.$$

Úlohu (P_h) můžeme zapsat také takto:

$$\left.\begin{aligned} &\text{Najdi } u_h(x) = \sum_{j=0}^{n+1} u_j \phi_j(x),\ u_0 = u_a,\ u_{n+1} = u_b \text{ tak, že} \\ &\sum_{j=0}^{n+1} u_j \int_a^b \phi_j' \phi_i' = \int_a^b f\phi_i,\ i = 1, \ldots, n. \end{aligned}\right\} (P_h')$$

Odtud snadno získáme soustavu lineárních rovnic

$$\mathbf{A}\boldsymbol{u} = \boldsymbol{b}, \tag{2}$$

kde $\mathbf{A} = (a_{ij}) \in \mathbb{R}^{n\times n}$, $\boldsymbol{b} = (b_i) \in \mathbb{R}^n$ a $\boldsymbol{u} = (u_i) \in \mathbb{R}^n$, přičemž:

$$\begin{aligned} a_{ij} &= \int_a^b \phi_j' \phi_i',\ i, j = 1, \ldots, n, \\ b_1 &= \int_a^b f\phi_1 - u_a \int_a^b \phi_0' \phi_1', \\ b_i &= \int_a^b f\phi_i,\ i = 2, \ldots, n-1, \\ b_n &= \int_a^b f\phi_n - u_b \int_a^b \phi_{n+1}' \phi_n'. \end{aligned}$$

Protože prvky a_{ij} matice $\mathbf{A}$ jsou nenulové pro $|i - j| \leq 1$, dostáváme matici s třídiagonální strukturou:

$$\begin{bmatrix} h_0^{-1} + h_1^{-1} & -h_1^{-1} & \cdots & 0 \\ -h_1^{-1} & h_1^{-1} + h_2^{-1} & -h_2^{-1} & \\ \vdots & \ddots & \ddots & \vdots \\ & & & -h_{n-1}^{-1} \\ 0 & \cdots & -h_{n-1}^{-1} & h_{n-1}^{-1} + h_n^{-1} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_{n-1} \\ u_n \end{bmatrix} = \begin{bmatrix} f(x_1)\frac{1}{2}(h_0 + h_1) + u_a h_0^{-1} \\ f(x_2)\frac{1}{2}(h_1 + h_2) \\ \vdots \\ f(x_{n-1})\frac{1}{2}(h_{n-2} + h_{n-1}) \\ f(x_n)\frac{1}{2}(h_{n-1} + h_n) + u_b h_n^{-1} \end{bmatrix}$$

Při sestavení vektoru $\boldsymbol{b}$ jsme použili numerickou integraci:

$$\int_a^b f\phi_i = \int_{x_{i-1}}^{x_{i+1}} f\phi_i \doteq f(x_i)\frac{1}{2}(h_{i-1} + h_i).$$

Uvedenou soustavu lineárních rovnic můžeme snadno zavést do počítače a vyřešit. Ukážeme

si však jiný postup pro sestavení matice $\mathbf{A}$ i vektoru $\boldsymbol{b}$, který lze snadno zobecnit pro vícedimenzionální úlohy a také vektorizovat.

2.3 Vektorizované sestavení lineární soustavy

Na druhém řádku úlohy (P'_h) budeme uvažovat všechny bázové funkce $\phi_i,\ i = 0, 1, \ldots, n+1$ a sestavíme nejprve rozšířenou soustavu lineárních rovnic bez okrajových podmínek:

$$\bar{\mathbf{A}}\bar{\boldsymbol{u}} = \bar{\boldsymbol{b}}, \tag{3}$$

kde $\bar{\mathbf{A}} \in \mathbb{R}^{(n+2)\times(n+2)}$, $\bar{\boldsymbol{b}} \in \mathbb{R}^{n+2}$ a $\bar{\boldsymbol{u}} \in \mathbb{R}^{n+2}$, která má tvar:

$$\begin{bmatrix} h_0^{-1} & -h_0^{-1} & \cdots & 0 \\ -h_0^{-1} & h_0^{-1}+h_1^{-1} & -h_1^{-1} & \\ \vdots & \ddots & \ddots & \vdots \\ & -h_{n-1}^{-1} & h_{n-1}^{-1}+h_n^{-1} & -h_n^{-1} \\ 0 & \cdots & -h_n^{-1} & h_n^{-1} \end{bmatrix} \begin{bmatrix} u_0 \\ u_1 \\ \vdots \\ u_n \\ u_{n+1} \end{bmatrix} = \begin{bmatrix} f(x_0)\frac{1}{2}h_0 \\ f(x_1)\frac{1}{2}(h_0+h_1) \\ \vdots \\ f(x_n)\frac{1}{2}(h_{n-1}+h_n) \\ f(x_{n+1})\frac{1}{2}h_n \end{bmatrix}$$

Odtud je vidět, že sestavení matice $\bar{\mathbf{A}}$ a vektoru $\bar{\boldsymbol{b}}$ lze provést vektorizovaným způsobem, který v MATLAB zapíšeme kódem 1.

```
% Assembly of the matrix
A=sparse(np,np); hinv=1./h;
A=A+sparse(t(:,1),t(:,1),hinv,np,np);
A=A+sparse(t(:,2),t(:,2),hinv,np,np);
A=A-sparse(t(:,1),t(:,2),hinv,np,np);
A=A-sparse(t(:,2),t(:,1),hinv,np,np);
% Assembly of the right-hand side vector
b=zeros(np,1);
b=b+sparse(t(:,1),1,(1/2)*f(p(1:end-1)).*h,np,1);
b=b+sparse(t(:,2),1,(1/2)*f(p(2:end)).*h,np,1);
```

Kód 1: Vektorizované sestavení lineární soustavy (3)

Uvedený způsob vektorizovaného sestavení matice $\bar{\mathbf{A}}$ a vektoru $\bar{\boldsymbol{b}}$ jsme navrhli ze znalosti vnitřní struktury těchto objektů. V dalším ukážeme odvození stejné vektorizace pomocí rozkladu cyklu, což je snadno přenositelné pro vícedimenzionální úlohy.

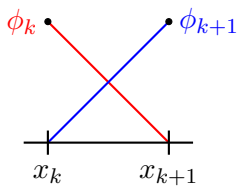
Prvky a_{ij} matice $\bar{\mathbf{A}}$ rozložíme pomocí aditivity integrálu takto:

$$a_{ij} = a_{ij0} + a_{ij1} + \ldots + a_{ijn}, \quad \text{kde } a_{ijk} = \int_{x_k}^{x_{k+1}} \phi'_j \phi'_i.$$

Matici $\bar{\mathbf{A}}$ můžeme tedy vygenerovat pomocí tří vnořených cyklů:

$$\begin{aligned}
&\bar{\mathbf{A}} := \mathbf{0};\\
&\texttt{for } i = 0, 1, \ldots, n+1\\
&\quad \texttt{for } j = 0, 1, \ldots, n+1\\
&\quad\quad \texttt{for } k = 0, 1, \ldots, n\\
&\quad\quad\quad a_{ij} := a_{ij} + a_{ijk};
\end{aligned}$$

Pořadí cyklů můžeme zaměnit tak, že vnější cyklus bude v proměnné k. Pro pevné k pracujeme na intervalu $T_k = \langle x_k, x_{k+1}\rangle$, kde jsou nenulové pouze bázové funkce ϕ_k a ϕ_{k+1}, viz. obrázek 3.



Obrázek 3: Bázové funkce na intervalu T_k

Vnitřní cykly (pro proměnné i a j) budou proto obsahovat pouze čtyři nenulové příspěvky do matice $\bar{\mathbf{A}}$ a to a_{kkk}, $a_{k+1k+1k}$, a_{k+1kk}, a_{kk+1k}, které tvoří tzv. lokální matici tuhosti:

$$\mathbf{A}_k = \begin{bmatrix} \int_{x_k}^{x_{k+1}} \phi_k'\phi_k' & \int_{x_k}^{x_{k+1}} \phi_{k+1}'\phi_k' \\ \int_{x_k}^{x_{k+1}} \phi_k'\phi_{k+1}' & \int_{x_k}^{x_{k+1}} \phi_{k+1}'\phi_{k+1}' \end{bmatrix} = \begin{bmatrix} h_k^{-1} & -h_k^{-1} \\ -h_k^{-1} & h_k^{-1} \end{bmatrix}.$$

Poslední vyjádření lokální matice tuhosti $\mathbf{A}_k$ vzniklo výpočtem jednotlivých integrálů. Pro sestavení globální matice $\bar{\mathbf{A}}$ dostáváme následující postup:

$$\begin{aligned}
&\bar{\mathbf{A}} = \mathbf{0};\\
&\texttt{for } k = 0, 1, \ldots, n\\
&\quad \begin{bmatrix} a_{kk} & a_{kk+1} \\ a_{k+1k} & a_{k+1k+1} \end{bmatrix} := \begin{bmatrix} a_{kk} & a_{kk+1} \\ a_{k+1k} & a_{k+1k+1} \end{bmatrix} + \begin{bmatrix} h_k^{-1} & -h_k^{-1} \\ -h_k^{-1} & h_k^{-1} \end{bmatrix};
\end{aligned}$$

Podobně sestavujeme také vektor pravé strany $\boldsymbol{b}$. Lokální vektor pravé strany má na intervalu $T_k = \langle x_n, x_{n+1}\rangle$ tvar:

$$\boldsymbol{b}_k = \begin{bmatrix} \int_{x_k}^{x_{k+1}} f\phi_k \\ \int_{x_k}^{x_{k+1}} f\phi_{k+1} \end{bmatrix} \doteq \begin{bmatrix} \frac{1}{2} f(x_k) h_k \\ \frac{1}{2} f(x_{k+1}) h_k \end{bmatrix}.$$

Celý vektor $\bar{\boldsymbol{b}}$ pak sestavíme takto:

$$\begin{aligned}&\bar{\boldsymbol{b}} := \mathbf{0};\\ &\texttt{for } k = 0, 1, \ldots, n\\ &\qquad \begin{bmatrix} b_k \\ b_{k+1} \end{bmatrix} := \begin{bmatrix} b_k \\ b_{k+1} \end{bmatrix} + \begin{bmatrix} \frac{1}{2} f(x_k) h_k \\ \frac{1}{2} f(x_{k+1}) h_k \end{bmatrix};\end{aligned}$$

V MATLABu těmto postupům odpovídá kód 2.

```
A=sparse(np,np); b=zeros(np,1);
for k=1:nt
    ind=t(k,:);
    % Element matrix
    A(ind,ind)=A(ind,ind)+hinv(k)*[1 -1;-1 1];
    % Element right-hand vector
    b(ind)=b(ind)+h(k)/2*[f(p(k));f(p(k+1))];
end
```

Kód 2: Sestavení matice $\bar{\mathbf{A}}$ a vektoru $\bar{\boldsymbol{b}}$ za pomocí cyklů

Cyklus pro sestavení matice $\bar{\mathbf{A}}$ pak můžeme zapsat, jako čtyři nezávislé cykly pro jednotlivé prvky lokální matice tuhosti:

$$\begin{aligned}&\bar{\mathbf{A}} = \mathbf{0};\\ &\texttt{for } k = 0, 1, \ldots, n\,,\ a_{kk} := a_{kk} + h_k^{-1};\\ &\texttt{for } k = 0, 1, \ldots, n\,,\ a_{k+1\,k} := a_{k+1\,k} - h_k^{-1};\\ &\texttt{for } k = 0, 1, \ldots, n\,,\ a_{k\,k+1} := a_{k\,k+1} - h_k^{-1};\\ &\texttt{for } k = 0, 1, \ldots, n\,,\ a_{k+1\,k+1} := a_{k+1\,k+1} + h_k^{-1};\end{aligned}$$

a cyklus pro sestavení vektoru $\bar{\boldsymbol{b}}$, jako dva nezávislé cykly:

$$\begin{aligned}&\bar{\boldsymbol{b}} = \mathbf{0};\\ &\texttt{for } k = 0, 1 \ldots, n,\ b_k := b_k + \frac{1}{2} f(x_k) h_k;\\ &\texttt{for } k = 0, 1 \ldots, n,\ b_{k+1} := b_{k+1} + \frac{1}{2} f(x_{k+1}) h_k;\end{aligned}$$

Těmto rozkladům cyklů pro matici $\bar{\mathbf{A}}$ i vektor $\bar{\boldsymbol{b}}$ odpovídá kód 1. Matici $\bar{\mathbf{A}}$ a vektor $\bar{\boldsymbol{b}}$ tedy umíme sestavit několika způsoby.

Nakonec ukážeme, jak do naší úlohy zahrnout okrajové podmínky. Víme, že hodnoty u_a, u_b odpovídají uzlům x_0, x_{n+1}, a proto položíme $\texttt{d} = [u_a, u_b]'$, $\texttt{dind} = [1, \texttt{np}]'$. Ze soustavy $\bar{\mathbf{A}}\bar{\boldsymbol{u}} = \bar{\boldsymbol{b}}$ vypustíme první a poslední rovnici. Následně z matice $\bar{\mathbf{A}}$ převedeme první a poslední sloupec

přenásobený vektorem `d` do vektoru pravé strany. Tímto postupem ze soustavy (3) vytvoříme soustavu (2). Tento postup v MATLABu zapíšeme kódem 3, který doplníme vyřešením soustavy (2) a opětovným zahrnutím okrajových do řešení.

```
% Boundary conditions
A(dind,:)=[]; b(dind)=[];
b=b-A(:,dind)*d;
A(:,dind)=[];
% Solve the linear system
u=A\b;
ind_all=(1:np)';
ind_minus=setdiff(ind_all,dind);
uu(ind_minus)=u;
% Problem solution
uu(dind)=d;
```

Kód 3: Zahrnutí okrajových podmínek a řešení soustavy

2.4 Transformace na referenční interval

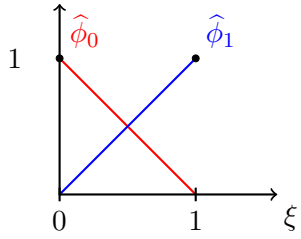
Všimněme si, že platí

$$\mathbf{A}_k = h_k^{-1} \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix},$$

kde matice na pravé straně nezávisí na konkrétním intervalu T_k. Můžeme ji proto sestavit na referenčním intervalu $\hat{T} = \langle 0, 1 \rangle$ pomocí refenčních bázových funkcí:

$$\hat{\phi}_0(\xi) = 1 - \xi, \;\; \hat{\phi}_1(\xi) = \xi,$$

viz obrázek 4.



Obrázek 4: Lineární bázové funkce na referenčním intervalu

Referenční matice tuhosti má tvar:

$$\mathbf{A}_{\text{ref}} = \begin{bmatrix} \int_0^1 \hat{\phi}_0' \hat{\phi}_0' & \int_0^1 \hat{\phi}_1' \hat{\phi}_0' \\ \int_0^1 \hat{\phi}_0' \hat{\phi}_1' & \int_0^1 \hat{\phi}_1' \hat{\phi}_1' \end{bmatrix} = \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix},$$

což lze zjistit snadným výpočtem. Přechod mezi intervalem T_k a referenčním intervalem $\widehat{T}$ zajišťuje lineární transformace

$$\xi_k : T_k \to \hat{T}, \quad \xi_k(x) = h_k^{-1}(x - x_k).$$

Snadno lze ukázat, že platí

$$\phi_k(x) = \hat{\phi}_0(\xi_k(x)), \quad \phi_{k+1}(x) = \hat{\phi}_1(\xi_k(x)).$$

Podle pravidla o derivování složené funkce dostaneme:

$$\phi_k'(x) = \hat{\phi}_0'(\xi_k(x))\xi_k'(x), \quad \phi_{k+1}'(x) = \hat{\phi}_1'(\xi_k(x))\xi_k(x).$$

Při substituci v integrálech z lokální matice tuhosti $\mathbf{A}_k$ využijeme vztahy:

$$\xi_k'(x) = h_k^{-1}, \quad dx = h_k \, d\xi_k.$$

Na ukázku naznačme substituci pro prvek a_{kkk} matice $\mathbf{A}_k$:

$$a_{kkk} = \int_{x_k}^{x_{k+1}} \phi_k'(x)\phi_k'(x) \, \mathrm{d}x = \int_0^1 \phi_0'(\xi_k) h_k^{-1} \phi_0'(\xi_k) h_k^{-1} h_k \, \mathrm{d}\xi_k = h_k^{-1} \int_0^1 \hat{\phi}_0'(\xi_k)\hat{\phi}_0'(\xi_k) \, \mathrm{d}\xi_k.$$

Dokázali jsme, že platí:

$$\mathbf{A}_k = h_k^{-1} \mathbf{A}_{\text{ref}}.$$

Analogii této rovnosti budeme používat u vícedimenzionálních úloh.

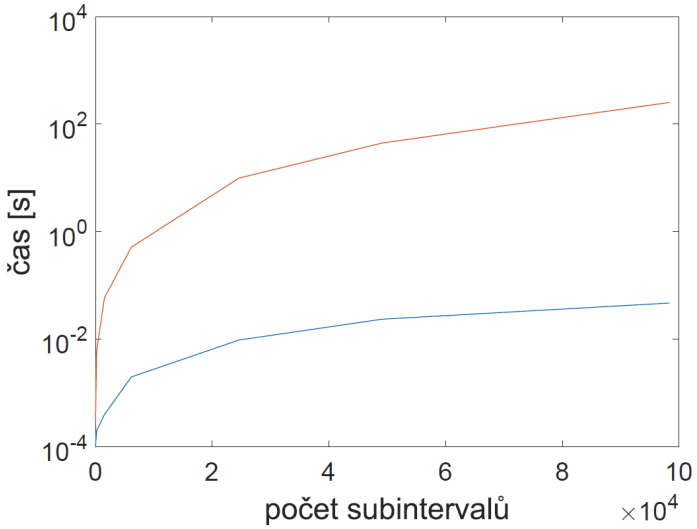
2.5 Výpočetní experimenty

Určení poměru časového vytížení CPU mezi standardním a vektorizovaným sestavením globální matice tuhosti a vektoru pravé strany bylo provedeno na domácí PC sestasvě s procesorem AMD FX-4300 Quad-Core - 3.80 GHz. Testované kódy byly měřeny zpravidla několikrát pro zpřesnění výsledků. Výsledky jsou zobrazeny v tabulce ??.

Tabulka 1: Výsledky časových experimentů

Počet intervalů	6	192	1536	6144	24576	49152	98304
Standardní sestavení (s)	0.0003	0.0061	0.0591	0.5171	9.91	44.666	252.82
Vektorizované sestavení (s)	0.0001	0.0002	0.0004	0.0020	0.0097	0.0237	0.0472
Poměr	47.42%	2.71%	0.716%	0.386%	0.099%	0.053%	0.019%

Pro lepší názornost je v tomto případě časová náročnost vyobrazena i graficky na obrázku 5. Z něj je patrné, že potřeba vykonat cyklus přes každý interval způsobuje, s jejich rostoucím počtem, značné výpočetní nároky. Naproti tomu vektorizované sestavení, kde k žádným cyklům nedochází, sice vykazuje určitou rostoucí tendenci, ale výsledná časová náročnost je ve srovnání se standardním sestavením při větším počtu intervalů výrazně menší.



Obrázek 5: Časová závislost sestavení s různým počtem subintervalů

3 Modelová úloha 2D

V této kapitole se budeme zabývat dvoudimenzionální skalární úlohou, která má podobný charakter jako úloha v kapitole 2. Opět navrhneme vektorizované kódy pro sestavení soustavy lineárních rovnic vzniklé z metody konečných prvků. Využijeme při tom zkušenosti z jednodimenzionální úlohy, takže již nebudeme muset zacházet příliš do detailů.

Také diferenciály u integrálů psát nebudeme vyjma případů, kdy jsou potřebné pro porozumění.

3.1 Formulace úlohy

Budeme uvažovat okrajovou úlohu pro parciální diferenciální rovnici druhého řádu. Nechť $\Omega \subset \mathbb{R}^2$ je omezená oblast s dostatečně hladkou hranicí $\partial\Omega$. Budeme předpokládat, že f je dostatečně hladká funkce definovaná na $\bar{\Omega}$ a g je dostatečně hladká funkce definovaná na $\partial\Omega$ představující Dirichletovu okrajovou podmínku a $\alpha > 0$ je daná konstanta. Hledáme funkci $u = u(x, y)$ na $\bar{\Omega}$ vyhovující následující úloze:

$$-\Delta u + \alpha u = f \text{ v } \Omega, \;\; u = g \text{ na } \partial\Omega. \tag{4}$$

Při odvozování slabé formulace použijeme Greenovu formuli a zeslabíme požadavky na hladkost řešení, viz dodatek A.2. Slabá formulace úlohy (4) má tvar:

$$\left.\begin{array}{l} \text{Najdi } u \in H^1(\Omega), \;\; u = g \text{ na } \partial\Omega \quad \text{tak, že} \\ \displaystyle\int_\Omega \nabla u \cdot \nabla v + \alpha u v = \int_\Omega f v \quad \forall v \in H_0^1(\Omega). \end{array}\right\} \; (P)$$

Zvolíme si síť uzlů $\mathcal{P} = \{p_i = [x_i, y_i]^\mathsf{T} \in \bar{\Omega} : \;\; i = 1, 2, \ldots, n\}$, které odpovídá regulární triangulace $\mathcal{T} = \{T = T_{ijk} : T_{ijk}$ je trojúhelník s vrcholy $\mathbf{p_i}, \; \mathbf{p_j}, \; \mathbf{p_k} \in \mathcal{P}\}$. Nadále budeme předpokládat, že Ω je polygonální oblast a triangulace $\mathcal{T}$ je s ní konzistentní, tj. $\bar{\Omega} = \bigcup\limits_{T \in \mathcal{T}} T$. Na triangulaci $\mathcal{T}$ budeme uvažovat lineární konečné prvky tak, že zavedeme následující aproximační množinu:

$$V_{hg} = \{v_h \in C(\bar{\Omega}) : v_{h|T} \in P_1(T) \; \forall T \in \mathcal{T}, v_{|\partial\Omega} = g\}.$$

Aproximace úlohy (P) metodou konečných prvků má tento tvar:

$$\left.\begin{array}{l} \text{Najdi } u_h \in V_{hg} \quad \text{tak, že} \\ \displaystyle\int_\Omega \nabla u_h \cdot \nabla v_h + \alpha u_h v_h = \int_\Omega f v_h \quad \forall v_h \in V_{h0}. \end{array}\right\} \; (P_h)$$

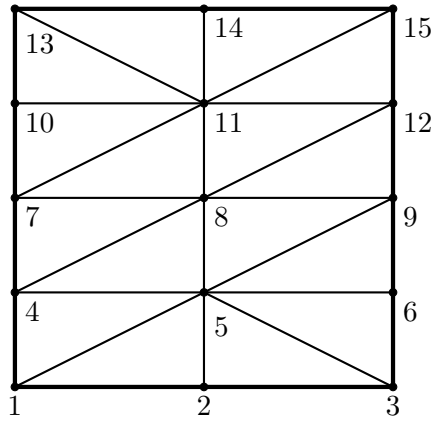
V MATLABu budeme síť $\mathcal{P}$ a triangulaci $\mathcal{T}$ reprezentovat pomocí polí `p` a `t` s touto strukturou:

$$\texttt{p} = [x_1, y_1; x_2, y_2; \ldots; x_n, y_n];\ \texttt{np} = \texttt{size}(\texttt{p}, 1);$$
$$\texttt{t} = [\ldots; i, j, k; \ldots];\ \texttt{nt} = \texttt{size}(\texttt{t}, 1);$$

kde i, j, k jsou indexy uzlů tvořících vrcholy trojúhelníku T_{ijk}.

Příklad 1

Na obrázku 6 uvádíme příklad triangulace čtverce $\bar{\Omega} = \langle 0, 1\rangle \times \langle 0, 1\rangle$ a vypisujeme část odpovídajících polí `p` a `t`. ■




```
p = [  0     0
     0.5     0
       1     0
       0  0.25
     0.5  0.25
       1  0.25
       0   0.5
     0.5   0.5
     ...     ]

np = 15
```

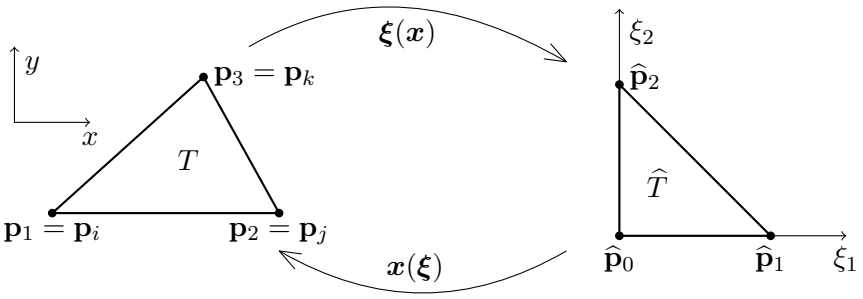
```
t = [  1   5   4
       1   2   5
       2   6   5
       2   3   6
       4   8   7
       4   5   8
       5   9   8
       5   6   9
     ...     ]

nt = 16
```

Obrázek 6: Zobrazení bodů a prvků sítě

Postup sestavení soustavy lineárních rovnic, na kterou vede úloha (P_h), navrhneme tak, aby byl nezávislý na pořadí uložení uzlů v `p` i na pořadí reprezentace trojúhelníků v `t`.

3.2 Transformace na referenční trojúhelník

Uvažujme trojúhelník $T = T_{ijk} \in \mathcal{T}$. Budeme používat lokální číslování uzlů: $\mathbf{p}_i = \mathbf{p}_1 = [x_1, y_1]^\mathsf{T}$ a podobně pro $\mathbf{p}_j = \mathbf{p}_2,\ \mathbf{p}_k = \mathbf{p}_3$, viz obrázek 7 vlevo.



Obrázek 7: Lokální číslování uzlů a referenční trojúhelník

Na trojúhelníku jsou nenulové tři lineární bázové funkce $\phi_1(\boldsymbol{x})$, $\phi_2(\boldsymbol{x})$, $\phi_3(\boldsymbol{x}) \in P_1(T)$, $\boldsymbol{x} = [x, y]^\mathsf{T} \in T$ definované podmínkami: $\phi_i(\mathbf{p}_j) = \delta_{ij}$, $i = 1, 2, 3$. Explicitní předpis pro tyto bázové funkce potřebovat nebudeme. Namísto toho pužijeme linéarní transformaci trojúhelníku T na referenční trojúhleník $\widehat{T}$ s vrcholy $\widehat{\mathbf{p}}_0 = [0, 0]^\mathsf{T}$, $\widehat{\mathbf{p}}_1 = [1, 0]^\mathsf{T}$ $\widehat{\mathbf{p}}_2 = [0, 1]^\mathsf{T}$, na kterém definujeme referenční bázové funkce:

$$\begin{aligned}
\hat{\phi}_0(\boldsymbol{\xi}) &= 1 - \xi_1 - \xi_2, \\
\hat{\phi}_1(\boldsymbol{\xi}) &= \xi_1, \\
\hat{\phi}_2(\boldsymbol{\xi}) &= \xi_2, \qquad \boldsymbol{\xi} = [\xi_1\, \xi_2]^\mathsf{T} \in \hat{T}.
\end{aligned}$$

Referenční bázové funkce budou sloužit také jako *barycentrické souřadnice* bodu $\boldsymbol{x} = [x, y]^\mathsf{T} \in T$ vzhledem k vrcholům trojúhelníku T:

$$\boldsymbol{x} = \mathbf{p}_1\widehat{\phi}_0(\boldsymbol{\xi}) + \mathbf{p}_2\widehat{\phi}_1(\boldsymbol{\xi}) + \mathbf{p}_3\widehat{\phi}_2(\boldsymbol{\xi}), \quad \boldsymbol{\xi} \in \widehat{T}.$$

Právě uvedený vztah je lineární transformací trojúhelníka $\widehat{T}$ na trojúhelník T:

$$\boldsymbol{x} = \boldsymbol{x}(\boldsymbol{\xi}) : \widehat{T} \to T. \tag{5}$$

Využijeme-li definici referenčních bázových funkcí, můžeme uvedenou transformaci zapsat takto:

$$\boldsymbol{x} = \mathbf{p}_1 + \mathbf{X}\boldsymbol{\xi}, \quad \mathbf{X} = [\mathbf{p}_2 - \mathbf{p}_1,\ \mathbf{p}_3 - \mathbf{p}_1] \in \mathbb{R}^{2\times 2}.$$

Determinant matice $\mathbf{X}$ můžeme snadno spočítat:

$$|\mathbf{X}| = \begin{vmatrix} x_2 - x_1 & x_3 - x_1 \\ y_2 - y_1 & y_3 - y_1 \end{vmatrix} = (x_2 - x_1)(y_3 - y_1) - (x_3 - x_1)(y_2 - y_1).$$

Je také zřejmé, že $|\mathbf{X}| = 2|T|$, kde $|T|$ je velikost trojúhelníku T.

Poznámka 1 Využíváme skutečnost, že vrcholy v trojúhelníku indexujeme proti směru hodinových ručiček. Jinak bychom pro určení velikosti trojúhelníku museli používat absolutí hodnotu.

Determinant $|\mathbf{X}|$ je proto nenulový pro každý nedegenerovaný trojúhelník a matice $\mathbf{X}$ je proto invertovatelná. K transformaci (5) můžeme zapsat inverzní transformaci:

$$\boldsymbol{\xi} = \boldsymbol{\xi}(\boldsymbol{x}) :\ T \to \widehat{T}$$

pomocí předpisu

$$\boldsymbol{\xi} = \mathbf{X}^{-1}(\boldsymbol{x} - \mathbf{p}_1). \tag{6}$$

Bázové funkce $\phi_1(\boldsymbol{x}),\ \phi_2(\boldsymbol{x}),\ \phi_3(\boldsymbol{x})$ lze pro $\boldsymbol{x} \in T$ zapsat následující substitucí:

$$\phi_1(\boldsymbol{x}) = \widehat{\phi}_0(\boldsymbol{\xi}(\boldsymbol{x})), \quad \phi_2(\boldsymbol{x}) = \widehat{\phi}_1(\boldsymbol{\xi}(\boldsymbol{x})), \quad \phi_3(\boldsymbol{x}) = \widehat{\phi}_2(\boldsymbol{\xi}(\boldsymbol{x})). \tag{7}$$

3.3 Sestavení soustavy lineárních rovnic

Podobně jako v případě 1D sestavíme k úloze (P_h) nejprve soustavu lineárních rovnic bez okrajových podmínek a budeme ji značit opět

$$\bar{\mathbf{A}}\bar{\boldsymbol{u}} = \bar{\boldsymbol{b}},$$

kde $\bar{\mathbf{A}} \in \mathbb{R}^{n\times n}$, $\bar{\boldsymbol{b}} \in \mathbb{R}^n$ a $\bar{\boldsymbol{u}} \in \mathbb{R}^n$. Poznamenejme, že uspořádání složek vektoru $\boldsymbol{u}$ bude odpovídat uspořádání jednotlivých uzlů v proměnné `p`. Narozdíl od situace v 1D však budeme o matici $\bar{\mathbf{A}}$ hovořit jako o *rozšířené matici tuhosti*, protože ji můžeme zapsat jako součet

$$\bar{\mathbf{A}} = \bar{\mathbf{R}} + \bar{\mathbf{M}},$$

kde $\bar{\mathbf{R}}$ je *matice tuhosti* odpovídající integrálu $\int_\Omega \nabla u_h \cdot \nabla v_h$ a $\bar{\mathbf{M}}$ je *matice hmotnosti* odpovídající integrálu $\int_\Omega \alpha u_h v_h$. Podobně budeme na trojúhelníku $T = T_{ijk} \in \mathcal{T}$ rozlišovat lokální matici tuhosti $\mathbf{R}_T \in \mathbb{R}^{3\times 3}$ a lokální matici hmotnosti $\mathbf{M}_T \in \mathbb{R}^{3\times 3}$, které společně tvoří rozšířenou lokální matici tuhosti:

$$\mathbf{A}_T = \mathbf{R}_T + \mathbf{M}_T.$$

Pro sestavení matice $\bar{\mathbf{M}}$ budeme potřebovat lokální maticí $\mathbf{M}_T$, která má tvar:

$$\mathbf{M}_T = \alpha \begin{bmatrix} \int_T \phi_1\phi_1 & \int_T \phi_2\phi_1 & \int_T \phi_3\phi_1 \\ \int_T \phi_1\phi_2 & \int_T \phi_2\phi_2 & \int_T \phi_3\phi_2 \\ \int_T \phi_1\phi_3 & \int_T \phi_2\phi_3 & \int_T \phi_3\phi_3 \end{bmatrix}.$$

Pro každý z integrálů provedeme substituci na referenční trojúhelník $\widehat{T}$. Například pro prvek (1,2) postupujeme takto:

$$\int_T \phi_2(\boldsymbol{x})\phi_1(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} = \int_{\widehat{T}} \phi_2(\boldsymbol{x}(\boldsymbol{\xi}))\phi_1(\boldsymbol{x}(\boldsymbol{\xi}))\ |\mathbf{X}|\,\mathrm{d}\boldsymbol{\xi} = 2|T| \int_{\widehat{T}} \widehat{\phi}_1(\boldsymbol{\xi})\widehat{\phi}_0(\boldsymbol{\xi})\,\mathrm{d}\boldsymbol{\xi}.$$

Lokální matici hmotnosti můžeme psát následovně:

$$\mathbf{M}_T = 2\alpha|T| \begin{bmatrix} \int_{\widehat{T}} \widehat{\phi}_0\widehat{\phi}_0 & \int_{\widehat{T}} \widehat{\phi}_1\widehat{\phi}_0 & \int_{\widehat{T}} \widehat{\phi}_2\widehat{\phi}_0 \\ \int_{\widehat{T}} \widehat{\phi}_0\widehat{\phi}_1 & \int_{\widehat{T}} \widehat{\phi}_1\widehat{\phi}_1 & \int_{\widehat{T}} \widehat{\phi}_2\widehat{\phi}_1 \\ \int_{\widehat{T}} \widehat{\phi}_0\widehat{\phi}_2 & \int_{\widehat{T}} \widehat{\phi}_1\widehat{\phi}_2 & \int_{\widehat{T}} \widehat{\phi}_2\widehat{\phi}_2 \end{bmatrix}. \tag{8}$$

Integrály z $\mathbf{M}_T$ můžeme vypočítat přímým výpočtem, numerickou integrací s dostatečným stupněm hladkosti nebo vhodným softwarem pro symbolický výpočet integrálu. Po provedení výpočtů všech integrálů, viz dodatky B.1, bude mít lokální matice hmotnosti následující tvar:

$$\mathbf{M}_T = \frac{\alpha|T|}{12} \begin{bmatrix} 2 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 2 \end{bmatrix}. \tag{9}$$

Úsek kódu v MATLABu, který sestaví globální matici hmotnosti s použitím cyklů, popisuje kód 4 a vektorizované sestavení kód 5.

```
M=sparse(np,np);
Mel=(1/12)*[2 1 1; 1 2 1; 1 1 2];
for k=1:nt
    ind=t(k,:);
    x21=p(ind(2),1)-p(ind(1),1); y12=p(ind(1),2)-p(ind(2),2);
    x13=p(ind(1),1)-p(ind(3),1); y31=p(ind(3),2)-p(ind(1),2);
    tarea=(x21*y31-x13*y12)/2;
    M(ind,ind)=M(ind,ind)+alpha*tarea*Mel;
end
```

Kód 4: Sestavení globální matice hmotnosti s použitím cyklů

```
M=sparse(np,np);
x21=p(t(:,2),1)-p(t(:,1),1); y12=p(t(:,1),2)-p(t(:,2),2);
x13=p(t(:,1),1)-p(t(:,3),1); y31=p(t(:,3),2)-p(t(:,1),2);
tarea=(x21.*y31-x13.*y12)/2; aux=alpha*tarea/12;
for i=1:3
   for j=1:i
      M=M+sparse(t(:,i),t(:,j), aux,np,np)+sparse(t(:,j),t(:,i), aux,np,np);
   end
end
```

Kód 5: Vektorizované sestavení globální matice hmotnosti

Nyní ukážeme jak sestavit matici $\bar{\mathbf{R}}$. Lokální matice tuhosti mají tento tvar:

$$\mathbf{R}_T = \begin{bmatrix} \int_T \nabla\phi_1 \cdot \nabla\phi_1 & \int_T \nabla\phi_2 \cdot \nabla\phi_1 & \int_T \nabla\phi_3 \cdot \nabla\phi_1 \\ \int_T \nabla\phi_1 \cdot \nabla\phi_2 & \int_T \nabla\phi_2 \cdot \nabla\phi_2 & \int_T \nabla\phi_3 \cdot \nabla\phi_2 \\ \int_T \nabla\phi_1 \cdot \nabla\phi_3 & \int_T \nabla\phi_2 \cdot \nabla\phi_3 & \int_T \nabla\phi_3 \cdot \nabla\phi_3 \end{bmatrix}.$$

Opět použijeme substituci na referenční trojúhelník $\widehat{T}$. Jak přitom pracovat s gradienty ukážeme pro bázovou funkci $\phi_1(\boldsymbol{x}) = \widehat{\phi}_0(\boldsymbol{\xi}(\boldsymbol{x}))$. Při výpočtu jejich parciálních derivací použijeme pravidlo pro derivaci složené funkce:

$$\frac{\partial\phi_1}{\partial x} = \frac{\partial\widehat{\phi}_0}{\partial\xi_1}\frac{\partial\xi_1}{\partial x} + \frac{\partial\widehat{\phi}_0}{\partial\xi_2}\frac{\partial\xi_2}{\partial x},$$
$$\frac{\partial\phi_1}{\partial y} = \frac{\partial\widehat{\phi}_0}{\partial\xi_1}\frac{\partial\xi_1}{\partial y} + \frac{\partial\widehat{\phi}_0}{\partial\xi_2}\frac{\partial\xi_2}{\partial y},$$

nebo-li

$$\left[\frac{\partial\phi_1}{\partial x}, \frac{\partial\phi_1}{\partial y}\right] = \left[\frac{\partial\widehat{\phi}_0}{\partial\xi_1}, \frac{\partial\widehat{\phi}_0}{\partial\xi_2}\right] \begin{bmatrix} \frac{\partial\xi_1}{\partial x} & \frac{\partial\xi_1}{\partial y} \\ \frac{\partial\xi_2}{\partial x} & \frac{\partial\xi_2}{\partial y} \end{bmatrix}.$$

Řádkové vektory jsou gradienty $\nabla\phi_1$ a $\nabla\widehat{\phi}_0$ a z (7) snadno zjistíme, že matice na pravé straně je $\mathbf{X}^{-1}$. Tuto inverzi vypočítáme Cramerovým pravidlem a dostaneme tento jednoduchý vzorec:

$$\mathbf{X}^{-1} = \frac{1}{2|T|}\begin{bmatrix} y_3 - y_1 & x_1 - x_3 \\ y_1 - y_2 & x_2 - x_1 \end{bmatrix}.$$

Uvedený postup lze provést také pro ϕ_2 a ϕ_3, takže platí:

$$\nabla\phi_1 = \nabla\widehat{\phi}_0\mathbf{X}^{-1},\ \nabla\phi_2 = \nabla\widehat{\phi}_1\mathbf{X}^{-1},\ \nabla\phi_3 = \nabla\widehat{\phi}_2\mathbf{X}^{-1}.$$

Snadno vypočítáme gradienty referenčních bázových funkcí:

$$\nabla\widehat{\phi}_0 = [-1,-1], \quad \nabla\widehat{\phi}_1 = [1,0], \quad \nabla\widehat{\phi}_2 = [0,1].$$

Odtud pro gradienty bázových funkcí na T dostaneme:

$$\nabla\phi_1 = \frac{1}{2|T|}[y_2 - y_3, x_3 - x_2], \quad \nabla\phi_2 = \frac{1}{2|T|}[y_3 - y_1, x_1 - x_3], \quad \nabla\phi_3 = \frac{1}{2|T|}[y_1 - y_2, x_2 - x_1].$$

Označíme-li

$$\boldsymbol{x}_T = \begin{bmatrix} y_2 - y_3 \\ y_3 - y_1 \\ y_1 - y_2 \end{bmatrix} = \begin{bmatrix} y_{23} \\ y_{31} \\ y_{12} \end{bmatrix}, \quad \mathbf{y}_T = \begin{bmatrix} x_3 - x_2 \\ x_1 - x_3 \\ x_2 - x_1 \end{bmatrix} = \begin{bmatrix} x_{32} \\ x_{13} \\ x_{21} \end{bmatrix},$$

můžeme psát

$$\left[\frac{\partial \phi_1}{\partial x}, \frac{\partial \phi_2}{\partial x}, \frac{\partial \phi_3}{\partial x}\right]^{\mathsf{T}} = \frac{1}{2|T|}\boldsymbol{x}_T, \quad \left[\frac{\partial \phi_1}{\partial y}, \frac{\partial \phi_2}{\partial y}, \frac{\partial \phi_3}{\partial y}\right]^{\mathsf{T}} = \frac{1}{2|T|}\mathbf{y}_T.$$

Pro prvky r_{ij} matice $\mathbf{R}_T$ dostáváme :

$$r_{ij} = \int_T \nabla\phi_i(\boldsymbol{x})\nabla\phi_j(\boldsymbol{x})d\boldsymbol{x} = \frac{1}{4|T|}(x_{Ti}x_{Tj} + y_{Ti}y_{Tj}), \ i,j = 1,2,3,$$

a celou matici $\mathbf{R}_T$ můžeme zapsat následovně:

$$\mathbf{R}_T = \frac{1}{4|T|}(\boldsymbol{x}_T\boldsymbol{x}_T^{\mathsf{T}} + \mathbf{y}_T\mathbf{y}_T^{\mathsf{T}}). \tag{10}$$

Úsek kóku v MATLABu generující matici $\bar{\mathbf{R}}$ průchodem přes trojúhelníky je uveden v kódu 6.

```
R=sparse(np,np);
% Loop over all triangles
for k=1:nt
    ind=t(k,:);
   % Triangles area
    x21=p(ind(2),1)-p(ind(1),1); y12=p(ind(1),2)-p(ind(2),2);
    x32=p(ind(3),1)-p(ind(2),1); y23=p(ind(2),2)-p(ind(3),2);
    x13=p(ind(1),1)-p(ind(3),1); y31=p(ind(3),2)-p(ind(1),2);
    tarea=(x21*y31-x13*y12)/2;
    xt=[y23;y31;y12]; yt=[x32;x13;x21];
   % Assembly of mass matrix
    R(ind,ind)=R(ind,ind)+(1/4/tarea)*(xt*xt'+yt*yt');
end
```

Kód 6: Sestavení globální matice tuhosti s použitím cyklů

Vektorizovaný úsek kódu představuje kód 7.

```
R=sparse(np,np);
x21=p(t(:,2),1)-p(t(:,1),1); y12=p(t(:,1),2)-p(t(:,2),2);
x32=p(t(:,3),1)-p(t(:,2),1); y23=p(t(:,2),2)-p(t(:,3),2);
x13=p(t(:,1),1)-p(t(:,3),1); y31=p(t(:,3),2)-p(t(:,1),2);
tarea=(x21.*y31-x13.*y12)/2; tau=0.25./tarea;
xt=[y23,y31,y12]; yt=[x32,x13,x21];
for i=1:3
   for j=1:3
      R=R+sparse(t(:,i),t(:,j),tau.*(xt(:,i).*xt(:,j),np,np)...
         +sparse(t(:,i),t(:,j),tau.*(yt(:,i).*yt(:,j),np,np);
   end
end
```

Kód 7: Vektorizované sestavení globální matice tuhosti

Lokální vektor pravé strany má tento tvar:

$$\boldsymbol{b}_T = \left[\int_T f\phi_1,\ \int_T f\phi_2,\ \int_T f\phi_3\ \right]^{\mathsf{T}}.$$

Použijeme numerickou integraci tak, že funkci f na T nahradíme průměrem funkčních hodnot z vrcholových bodů trojúhelníku T, tj,

$$f_T = \frac{1}{3}(f(\mathbf{p}_1) + f(\mathbf{p}_2) + f(\mathbf{p}_3)).$$

Dostáváme

$$\int_T f\phi(\boldsymbol{x})_i \mathrm{d}\,\boldsymbol{x} \doteq f\int_T \phi_i(\boldsymbol{x})\mathrm{d}\,\boldsymbol{x} = \frac{1}{3}|T|f_T,\ i = 1,2,3,$$

a proto

$$\boldsymbol{b}_T = \frac{1}{3}|T|f_T\,[1,\ 1,\ 1]^{\mathsf{T}}. \tag{11}$$

V MATLABu můžeme napsat tento nevektorizovaný kód:

```
b=zeros(np,1);
ft=(f(p(t(:,1),:))+f(p(t(:,2),:))+f(p(t(:,3),:)))/3;
aux=(1/3)*tarea*[1;1;1];
for k=1:nt
    ind=t(k,:);
    b(ind)=b(ind)+ft(k)*aux;
end
```

Kód 8: Sestavení vektoru pravé strany s použitím cyklů

Obdobně jako v předchozích případech lze tento kód zapsat také vektorizovaně, viz kod 9.

```
b=zeros(np,1);
ft=(f(p(t(:,1),:))+f(p(t(:,2),:))+f(p(t(:,3),:)))/3;
aux=(1/3)*ft.*tarea;
% Assembly of righ-hand side vector
for i=1:3
   b=b+sparse(t(:,i),1,aux,np,1);
end
```

Kód 9: Vektorizované sestavení vektoru pravé strany

Nakonec ukážeme, jak do úlohy zahrnout okrajové podmínky a celou úlohu vyřešit. Budeme předpokládat, že proměnná `dind` obsahuje indexy všech hraničních uzlů. Nejprve vypočteme Dirichletovy podmínky pro hraniční uzly a součtem matic $\bar{\mathbf{R}}$ a $\bar{\mathbf{M}}$ získáme rozšířenou matici tuhosti $\bar{\mathbf{A}}$. Dále je už postup totožný jako v 1D, viz kód 10.

```
% Matrix from a stiffness and mass matrices
A=R+M;
% Boundary conditions
d=g(p(dind,1),p(dind,2));
A(dind,:)=[]; b(dind)=[];
b=b-A(:,dind)*d;
A(:,dind)=[];
% Solve the linear system
u=A\b;
ind_all=(1:np)';
ind_minus=setdiff(ind_all,dind);
uu(ind_minus)=u;
% Problem solution
uu(dind)=d;
```

Kód 10: Realizace okrajových podmínek a vyřešení lineární soustavy

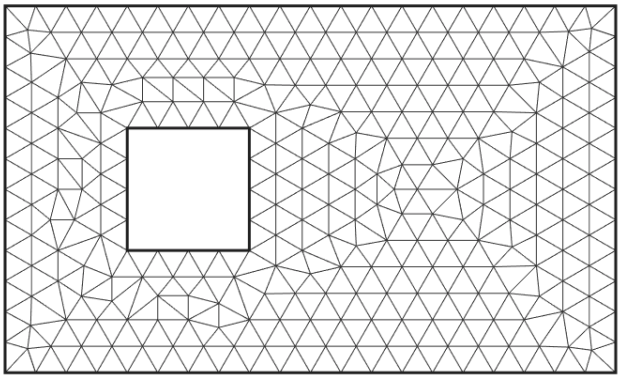
Do řešení soustavy `u` je následně přidáno řešení bodů z Dirichletových podmínek. Vektor `uu` pak představuje výsledné řešení úlohy (P_h).

3.4 Výpočetní experimenty

Byly provedeny obdobné experimenty jako v 1D případě. Tyto experimenty proběhly pro sedm útvarů, přičemž pro názornost jsou zobrazeny pouze dva, viz obrázek 8 a 9. Výsledky experimentů jsou zobrazeny v tabulkách 2 a 3.

Tabulka 2: Výsledky časových experimentů pro první oblast Ω

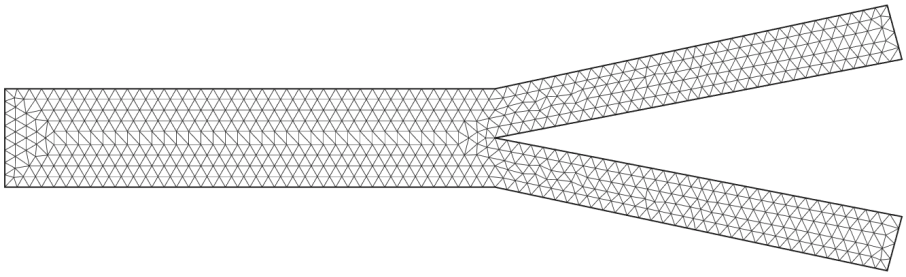
Počet trojúhelníků	3012	5902	11830	23518	47160	94806	189228
Standardní sestavení (s)	0.2139	0.5376	1.911	7.961	37.04	179.2	909.2
Vektorizované sestavení (s)	0.0108	0.0205	0.042	0.086	0.217	0.466	1.061
Poměr	5.049%	3.813%	2.19%	1.08%	0.58%	0.26%	0.12%



Obrázek 8: Síť pro první oblast Ω

Tabulka 3: Výsledky časových experimentů pro druhou oblast Ω

Počet trojúhelníků	2773	5510	11095	22292	44343	88978	177476
Standardní sestavení (s)	0.1922	0.5112	1.733	7.373	33.84	160.5	764.4
Vektorizované sestavení (s)	0.0106	0.0209	0.041	0.084	0.213	0.458	0.984
Poměr	5.515%	4.088%	2.36%	1.14%	0.63%	0.29%	0.13%



Obrázek 9: Síť pro druhou oblast Ω

Ze srovnání časů z obou oblastí je patrné, že standardní sestavení vyžaduje značné výpočetní nároky, zatímco náročnost vektorizovaného sestavení sice roste, ale jak je patrné ze srovnání v posledním řádku v tabulce 2 a 3, je tento růst značně menší. Dále je patrné, že pro větší hustotu sítí, kdy už většinu času zabírá samotné sestavení, nikoliv inicializace programu atd. jsou časové poměry nezávislé na tvaru, ale pouze na velikosti sítě.

3.5 Rozšíření aproximace o bublinkové funkce

Uvažujme modelovou úlohu (4) a její slabou formulací (P), síť uzlů $\mathcal{P}$ a triangulací $\mathcal{T}$, které byly definována v kapitole 3.1. Současnou aproximaci množinou V_{hg} nyní obohatíme o bublinkové funkce, které jsou na každém trojúhelníku $T \in \mathcal{T}$ definovány takto:

$$\phi_b(\boldsymbol{x}) = \phi_{Tb}(\boldsymbol{x}) = 3^3\phi_1(\boldsymbol{x})\phi_2(\boldsymbol{x})\phi_3(\boldsymbol{x}), \quad \boldsymbol{x} \in T,$$

kde $\phi_1,\ \phi_2,\ \phi_3$ jsou standardní lineární bázové funkce na T (používáme lokální číslování). Zaveďme tedy prostor bublinkových funkcí:

$$B_h = \{v_h \in C(\bar{\Omega}) : v_{h|T} = k_T\,\phi_{Tb} \quad \forall T \in \mathcal{T},\ k_T \in \mathbb{R}\}.$$

Budeme se tedy zabývat původní aproximací úlohy (P) rozšířenou o prostor B_h, která má tvar:

$$\left.\begin{array}{l}\text{Najdi } u_h \in V_{hg} \oplus B_h \text{ tak, že} \\ \displaystyle\int_\Omega \nabla u_h \cdot \nabla v_h + \alpha u_h v_h = \int_\Omega f v_h \quad \forall v_h \in V_{h0} \oplus B_h\end{array}\right\} (P_h^b).$$

Lemma 1 *Platí*

$$\nabla\phi_1(\boldsymbol{x}) + \nabla\phi_2(\boldsymbol{x}) + \nabla\phi_3(\boldsymbol{x}) = 0 \quad \forall \boldsymbol{x} \in T.$$

Důkaz

Plyne ze vztahu $\phi_1(\boldsymbol{x}) + \phi_2(\boldsymbol{x}) + \phi_3(\boldsymbol{x}) = 1$ ∎

Na $\widehat{T}$ přibude k referenčním lineárním bázovým funkcím $\widehat{\phi}_0,\ \widehat{\phi}_1,\ \widehat{\phi}_2$ referenční bublinková funkce:

$$\widehat{\phi}_b(\boldsymbol{\xi}) = 27\widehat{\phi}_0(\boldsymbol{\xi})\widehat{\phi}_1(\boldsymbol{\xi})\widehat{\phi}_2(\boldsymbol{\xi}), \quad \boldsymbol{\xi} \in \widehat{T}.$$

3.6 Sestavení soustavy s bublinkovými funkcemi

Ukážeme si, jak k úloze (P_h^b) sestavit odpovídající soustavu lineárních rovnic:

$$\bar{\mathbf{A}}\bar{\boldsymbol{u}} = \bar{\boldsymbol{b}}, \tag{12}$$

kde $\bar{\mathbf{A}} \in \mathbb{R}^{n\times n}$, $\bar{\boldsymbol{b}} \in \mathbb{R}^n$ a $\bar{\boldsymbol{u}} \in \mathbb{R}^n$. Řád této soustavy neodpovídá počtu bázových funkcí, kterých je $n+n_t$, kde n_t je počet trojúhelníků. Je to způsobeno tím, že neznámé odpovídající bublinkovým funkcím vyeliminujeme na lokální úrovni a na globální úrovni s nimi nebudeme vůbec pracovat.

Potřebujeme určit, jak sestavit celkovou lokální *matici tuhosti* a *matici hmostnosti* včetně bublinkových funkcí. Tyto matice jsou čtvrtého řádu a budeme je značit $\mathbf{R}_T^b \in \mathbb{R}^{4\times4}$ a $\mathbf{M}_T^b \in \mathbb{R}^{4\times4}$. Veškeré integrály byly počítány ručně nebo s použitím toolboxu SYMBOLIC v MATLABu, viz B.1.

Nejdříve se budeme věnovat matici $\mathbf{M}_T^b$, pro níž můžeme psát

$$\mathbf{M}_T^b = \begin{bmatrix} \mathbf{M}_T & \boldsymbol{m}_T \\ \boldsymbol{m}_T^\mathsf{T} & \omega_M \end{bmatrix},$$

kde $\mathbf{M}_T \in \mathbb{R}^{3\times 3}$ popisuje vztah (9). Bloky $\boldsymbol{m}_T \in \mathbb{R}^3$ a $\omega_T \in \mathbb{R}$ musíme odvodit:

$$\boldsymbol{m}_T = \left[\alpha \int_T \phi_b\phi_1,\ \alpha \int_T \phi_b\phi_2,\ \alpha \int_T \phi_b\phi_3\right]^\mathsf{T}, \qquad \omega_M = \alpha \int_T \phi_b\phi_b.$$

Opět použijeme substituci na referenční trojúhelník $\widehat{T}$ s referenčními bázovými funkcemi. Pak můžeme například pro prvek $(\boldsymbol{m}_T)_1$ psát:

$$(\boldsymbol{m}_T)_1 = \alpha \int_T \phi_b(\boldsymbol{x})\phi_1(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} = 54\alpha|T| \int_{\widehat{T}} \widehat{\phi}_0(\boldsymbol{\xi})^2\widehat{\phi}_1(\boldsymbol{\xi})\widehat{\phi}_2(\boldsymbol{\xi})\,\mathrm{d}\boldsymbol{\xi} = \frac{3\alpha|T|}{20}.$$

Podobně odvodíme $(\boldsymbol{m}_T)_2 = (\boldsymbol{m}_T)_3 = \frac{3\alpha|T|}{20}$, tak že

$$\boldsymbol{m}_T = \frac{3\alpha|T|}{20}\,[1,1,1]^\mathsf{T}.$$

Dále pro ω_M platí:

$$\omega_M = 2\alpha|T| \int_{\widehat{T}} \widehat{\phi}_b(\boldsymbol{\xi})\widehat{\phi}_b(\boldsymbol{\xi})\,\mathrm{d}\boldsymbol{\xi} = 2\cdot 27^2\alpha|T| \int_{\widehat{T}} \widehat{\phi}_0(\boldsymbol{\xi})^2\widehat{\phi}_1(\boldsymbol{\xi})^2\widehat{\phi}_2(\boldsymbol{\xi})^2\,\mathrm{d}\boldsymbol{\xi} = \frac{81\alpha|T|}{280}.$$

Pokračujme s *maticí tuhosti*:

$$\mathbf{R}_T^b = \begin{bmatrix} \mathbf{R}_T & \boldsymbol{r}_T \\ \boldsymbol{r}_T^\mathsf{T} & \omega_R \end{bmatrix},$$

kde lokální matice tuhosti bez bublinkových funkcí $\mathbf{R}_T \in \mathbb{R}^{3\times 3}$ je popsána vztahem (10). Blok $\boldsymbol{r}_T \in \mathbb{R}^3$ má tvar:

$$\boldsymbol{r}_T = \left[\alpha \int_T \nabla\phi_b\nabla\phi_1, \quad \alpha \int_T \nabla\phi_b\nabla\phi_2, \quad \alpha \int_T \nabla\phi_b\nabla\phi_3\right]^\mathsf{T}.$$

V tomto případě budeme substituci na $\widehat{T}$ potřebovat jen částečně, protože například:

$$\begin{aligned}(\boldsymbol{r}_T)_1 &= 27 \int_T \nabla(\phi_1(\boldsymbol{x})\phi_2(\boldsymbol{x})\phi_3(\boldsymbol{x}))\cdot\nabla\phi_1(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} = \\ &= 27\left(\int_T \phi_2(\boldsymbol{x})\phi_3(\boldsymbol{x})\nabla\phi_1(\boldsymbol{x})\cdot\nabla\phi_1(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} + \right. \\ &\quad + \int_T \phi_1(\boldsymbol{x})\phi_3(\boldsymbol{x})\nabla\phi_2(\boldsymbol{x})\cdot\nabla\phi_1(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} + \\ &\quad \left. + \int_T \phi_1(\boldsymbol{x})\phi_2(\boldsymbol{x})\nabla\phi_3(\boldsymbol{x})\cdot\nabla\phi_1(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x}\right).\end{aligned}$$

Skalární součiny gradientů jsou konstantní, takže je můžeme napsat před integrály. Například

pro první dílčí integrál dostaneme:

$$\begin{aligned}\int_T \phi_2(\boldsymbol{x})\phi_3(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} &= 2|T|\int_{\widehat{T}} \widehat{\phi}_1(\boldsymbol{\xi})\widehat{\phi}_2(\boldsymbol{\xi})\,\mathrm{d}\boldsymbol{\xi} = 2|T|\int_0^1\int_0^{1-\xi_1} \xi_1\xi_2\,\mathrm{d}\xi_2\,\mathrm{d}\xi_1 = \\ &= 2|T|\int_0^1 \frac{\xi_1(1-\xi_1)^2}{2}\,\mathrm{d}\xi_1 = |T|\int_0^1 \xi_1 - 2\xi_1^2 + \xi_1^3\,\mathrm{d}\xi_1 = \\ &= |T|\left[\frac{\xi_1^2}{2} - \frac{2\xi_1^3}{3} + \frac{\xi_1^4}{4}\right]_0^1 = |T|\left(\frac{1}{2} - \frac{2}{3} + \frac{1}{4}\right) = \frac{|T|}{12}.\end{aligned}$$

Podobně lze určit: $\int_T \phi_1(\boldsymbol{x})\phi_3(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} = \int_T \phi_1(\boldsymbol{x})\phi_2(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} = \frac{|T|}{12}$. Dohromady pak dostaneme:

$$\begin{aligned}(\mathbf{r}_T)_1 &= \frac{9|T|}{4}\left(\nabla\phi_1(\boldsymbol{x})\cdot\nabla\phi_1(\boldsymbol{x}) + \nabla\phi_2(\boldsymbol{x})\cdot\nabla\phi_3(\boldsymbol{x}) + \nabla\phi_3(\boldsymbol{x})\cdot\nabla\phi_1(\boldsymbol{x})\right) = \\ &= \frac{9|T|}{4}\left(\nabla\phi_1(\boldsymbol{x}) + \nabla\phi_2(\boldsymbol{x}) + \nabla\phi_3(\boldsymbol{x})\right)\cdot\nabla\phi_1(\boldsymbol{x}) = 0, \qquad (13)\end{aligned}$$

kde jsme použili Lemma 1. Podobně odvodíme $(\boldsymbol{r}_T)_2 = (\boldsymbol{r}_T)_3 = 0$, tak že $\boldsymbol{r}_T = \mathbf{0}$. Při výpočtu w_R postupujeme podobně:

$$\begin{aligned}\omega_R &= \int_T \nabla\phi_b\cdot\nabla\phi_b\,\mathrm{d}\boldsymbol{x} = 27^2\int_T \nabla(\phi_1\phi_2\phi_3)\cdot\nabla(\phi_1\phi_2\phi_3)\,\mathrm{d}\boldsymbol{x} = \\ &= 27^2\left[(\nabla\phi_1\cdot\nabla\phi_1)\int_T \phi_2^2\phi_3^2\,\mathrm{d}\boldsymbol{x} + (\nabla\phi_2\cdot\nabla\phi_2)\int_T \phi_1^2\phi_3^2\,\mathrm{d}\boldsymbol{x} + (\nabla\phi_3\cdot\nabla\phi_3)\int_T \phi_1^2\phi_2^2\,\mathrm{d}\boldsymbol{x} + \right. \\ &\left. +2(\nabla\phi_1\cdot\nabla\phi_2)\int_T \phi_1\phi_2\phi_3^2\,\mathrm{d}\boldsymbol{x} + 2(\nabla\phi_1\cdot\nabla\phi_3)\int_T \phi_1\phi_2^2\phi_3\,\mathrm{d}\boldsymbol{x} + 2(\nabla\phi_2\cdot\nabla\phi_3)\int_T \phi_1^2\phi_2\phi_3\,\mathrm{d}\boldsymbol{x}\right].\end{aligned}$$

Výpočtem s pomocí substituce na referenční trojúhelník $\widehat{T}$ dostaneme:

$$\int_T \phi_2^2\phi_3^2\,\mathrm{d}\boldsymbol{x} = \int_T \phi_1^2\phi_3^2\,\mathrm{d}\boldsymbol{x} = \int_T \phi_1^2\phi_2^2\,\mathrm{d}\boldsymbol{x} = \frac{|T|}{90},$$

$$\int_T \phi_1\phi_2\phi_3^2\,\mathrm{d}\boldsymbol{x} = \int_T \phi_1\phi_2^2\phi_3\,\mathrm{d}\boldsymbol{x} = \int_T \phi_1^2\phi_2\phi_3\,\mathrm{d}\boldsymbol{x} = \frac{|T|}{180}.$$

Dále

$$\begin{aligned}\omega_R &= \frac{27^2|T|}{90}((\nabla\phi_1\cdot\nabla\phi_1)+(\nabla\phi_2\cdot\nabla\phi_2)+(\nabla\phi_3\cdot\nabla\phi_3)+\\ &+(\nabla\phi_1\cdot\nabla\phi_2)+(\nabla\phi_1\cdot\nabla\phi_3)+(\nabla\phi_2\cdot\nabla\phi_3))=\\ &= \frac{81|T|}{10}((\nabla\phi_1\cdot\nabla\phi_1)+(\nabla\phi_2\cdot\nabla\phi_2)+(\nabla\phi_1\cdot\nabla\phi_2))=\\ &= \frac{81|T|}{10}(\nabla\phi_1\cdot\nabla\phi_1-\nabla\phi_2\cdot\nabla\phi_3)=\\ &= \frac{81}{40|T|}(x_{T1}^2+y_{T1}^2-x_{T2}x_{T3}-y_{T2}y_{T3}),\end{aligned}$$

kde jsme dvakrát použili Lemma 1.

Nakonec ukážeme jak vypadá lokální vektor pravé strany $\boldsymbol{b}_T^b \in \mathbb{R}^4$:

$$\boldsymbol{b}_T^b = \begin{bmatrix} \boldsymbol{b}_T \\ b_b \end{bmatrix},$$

kde vektor $\boldsymbol{b}_T \in \mathbb{R}^3$ je definován vztahem (11) a

$$b_b = \int_T f\phi_b \,\mathrm{d}\boldsymbol{x} \doteq 2|T| f_T \int_{\widehat{T}} \widehat{\phi}_b(\boldsymbol{\xi})\,\mathrm{d}\boldsymbol{\xi} = \frac{9}{20}|T| f_T.$$

Na lokálni úrovni, tj. na trojúhelníku T, uvažujeme lokální soustavu lineárních rovnic

$$\begin{bmatrix} \mathbf{A}_T & \boldsymbol{m}_T \\ \boldsymbol{m}_T^\mathsf{T} & \omega \end{bmatrix} \begin{bmatrix} \boldsymbol{u}_T \\ u_b \end{bmatrix} = \begin{bmatrix} \boldsymbol{b}_T \\ b_b \end{bmatrix}, \tag{14}$$

kde $\mathbf{A}_T = \mathbf{R}_T + \mathbf{M}_T$, $\omega = \omega_R + \omega_m$, $\boldsymbol{u}_T \in \mathbb{R}^3$, $u_b \in \mathbb{R}$. Ze druhé rovnice v (14) vyjádříme druhou neznámou:

$$u_b = \omega^{-1}(b_b - \boldsymbol{m}_T^\mathsf{T}\boldsymbol{u}_T),$$

a dosadíme do první rovnice v (14):

$$(\mathbf{A}_T - \omega^{-1}\boldsymbol{m}_T^\mathsf{T}\boldsymbol{m}_T)\boldsymbol{u}_T = \boldsymbol{b}_T - \omega^{-1}b_b\boldsymbol{m}_T.$$

Matici a vektor pravé strany této soustavy označíme

$$\mathbf{A}_T^b = \mathbf{A}_T - \omega^{-1}\boldsymbol{m}_T^\mathsf{T}\boldsymbol{m}_T, \qquad \boldsymbol{b}_T^b = \boldsymbol{b}_T - \omega^{-1}b_b\boldsymbol{m}_T.$$

Objekty $\mathbf{A}_T^b$ a $\boldsymbol{b}_T^b$ budeme vytvářet na lokální úrovni a následně z nich obvyklým způsobem sestavíme objekty globální. Na základě provedeného odvozování můžeme napsat MATLABovský úsek kódu, kterým sestavíme matici $\bar{\mathbf{A}}$ a vektor $\bar{\boldsymbol{b}}$ průchodem přes jednotlivé trojúhelníky. Kód 11 je sestaven z úseků kódů odvozených v kapitole 3.3 a dále je rozšířen o členy reprezentující bublinkové funkce.

```
% Assembly of the matrix the right-hand side
A=sparse(np,np); b=zeros(np,1);
Mel=(1/12)*[2 1 1; 1 2 1; 1 1 2];
% Loop over all triangles
for k=1:nt
    ind=t(k,:);
    % Triangles area
    x21=p(ind(2),1)-p(ind(1),1); y12=p(ind(1),2)-p(ind(2),2);
    x32=p(ind(3),1)-p(ind(2),1); y23=p(ind(2),2)-p(ind(3),2);
    x13=p(ind(1),1)-p(ind(3),1); y31=p(ind(3),2)-p(ind(1),2);
    tarea=(x21*y31-x13*y12)/2;
    xt=[y23;y31;y12]; yt=[x32;x13;x21];
    omega=81/40*(xt(1)^2+yt(1)^2-xt(2)*xt(3)-yt(2)*yt(3))/tarea...
                                                    +81/280*alpha*tarea;
    mt=3/20*alpha*tarea*[1;1;1];
    ft=(fp(ind(1))+fp(ind(2))+fp(ind(3)))/3;
    % Element matrix and vector with P1-bubble
    A(ind,ind)=A(ind,ind)+alpha*tarea*Mel+(1/4/tarea)...
                          *(xt*xt'+yt*yt')-mt*mt'/omega;
    b(ind)=b(ind)+ft*tarea*((1/3)*[1;1;1]-(9/20)/omega*mt);
end
```

Kód 11: Sestavení globálních objektů s použitím cyklů

Dále lze tento kód obdobně, jako v předchozích případech, zapsat vektorizovaně. Toto sestavení soustavy (12) vychází z úseků kódů odvozených v kapitole 3.3 a dále je rozšířeno o členy vyjadřující bublinkové funkce.

```
A=sparse(np,np); b=zeros(np,1);
x21=p(t(:,2),1)-p(t(:,1),1); y12=p(t(:,1),2)-p(t(:,2),2);
x32=p(t(:,3),1)-p(t(:,2),1); y23=p(t(:,2),2)-p(t(:,3),2);
x13=p(t(:,1),1)-p(t(:,3),1); y31=p(t(:,3),2)-p(t(:,1),2);
tarea=(x21.*y31-x13.*y12)/2; aux=alpha*tarea/12;
xt=[y23,y31,y12]; yt=[x32,x13,x21]; tau=0.25./tarea;
ft=(fp(t(:,1))+fp(t(:,2))+fp(t(:,3)))/3; mt=3*alpha*tarea/20;
omega=81/40*(xt(:,1).^2+yt(:,1).^2-xt(:,2).*xt(:,3)-yt(:,2).*yt(:,3))./tarea...
                                                    +81*alpha*tarea/280;
bux=(1/3)*ft.*tarea-9/20*ft.*tarea.*mt./omega;
for i=1:3
   % Mass matrix increment
   for j=1:i
      A=A+sparse(t(:,i),t(:,j),aux,np,np)...
          +sparse(t(:,j),t(:,i),aux,np,np);
   end
   % Stiffness matrix increment
   for j=1:3
      A=A+sparse(t(:,i),t(:,j),tau.*(xt(:,i).*xt(:,j)+yt(:,i).*yt(:,j))...
        -mt.*mt./omega ,np,np);
   end
   % Right side vector increment
   b=b+sparse(t(:,i),1,bux,np,1);
end
```

Kód 12: Vektorizované sestavení globálních objektů

Výpočetními experimenty na velkých soustavách, byla srovnána CPU náročnost vektorizovaného sestavení bez a s bublinkových funkcí, viz tabulka 4. S ohledem na naměřené časy lze vypozorovat, že časová náročnost sestavení s bublinkovými funkcemi je větší o 3-5 %.

Tabulka 4: Časy pro vektorové sestavení větších soustav

Počet elementů	85254	169608	340992	681186	1363882	2723936	5447102
Sestavení bez bubl. (s)	0.4225	1.014	2.152	4.631	9.766	20.31	40.24
Sestavení s bubl. (s)	0.4425	1.048	2.236	4.788	10.16	21.09	41.68

4 Modelová úloha 3D

V této kapitole si ukážeme třídimenzionální skalární úlohu, která vznikla rozšířením 2D úlohy z předchozí kapitoly, o další dimenzi a poté odvodíme kódy pro vektorizované sestavení soustavy lineárních rovnic vzniklé z metody konečných prvků. Budeme přitom vycházet ze znalostí uplatněných v jednodimenzionální a dvoudimenzionální úloze.

4.1 Formulace úlohy

Uvažujme okrajovou úlohu pro parciální diferenciální rovnici druhého řádu ve třech dimenzích. Nechť $\Omega \subset \mathbb{R}^3$ je omezená oblast s dostatečně hladkou hranicí $\partial\Omega$. Znovu budeme předpokládat, že f je dostatečně hladká funkce definovaná na $\bar{\Omega}$ a g je dostatečně hladká funkce definovaná na $\partial\Omega$ představující Dirichletovu okrajovou podmínku a $\alpha > 0$ je daná konstanta. Hledáme funkci $u = u(x, y, z)$ na $\bar{\Omega}$ vyhovující následující úloze:

$$-\Delta u + \alpha u = f \quad \text{v } \Omega,\ u = g \text{ na } \partial\Omega, \tag{15}$$

kde g jsou předepsané Dirichletovy podmínky. Odvozená slabá formulace i její aproximace je formálně stejná jako ve 2D případě. Má tedy tvar:

$$\left.\begin{array}{l} \text{Najdi } u \in H^1(\Omega),\ u = g \text{ na } \partial\Omega \quad \text{tak, že} \\ \displaystyle\int_\Omega \nabla u \cdot \nabla v + \alpha uv = \int_\Omega fv \quad \forall v \in H_0^1(\Omega). \end{array}\right\} \ (P)$$

Zvolíme síť uzlů $\mathcal{P} = \{\mathbf{p}_i = [x_i, y_i, z_i]^\mathsf{T} \in \bar{\Omega};\ i = 1, 2, \ldots, n\}$ s dělením na regulární čtyřstěny $\mathcal{T} = \{T = T_{ijkl}\ : T_{ijkl}$ je čtyřstěn s vrcholy $\mathbf{p}_i, \mathbf{p}_j, \mathbf{p}_k, \mathbf{p}_l \in \mathcal{P}\}$ takové, že $\bar{\Omega} = \bigcup_{T\in\mathcal{T}} T$. Opět předpokládáme, že Ω je polygonální oblast a dělení je konzistentní. Na dělení $\mathcal{T}$ budeme uvažovat lineární konečné prvky:

$$V_{hg} = \{v_h \in C(\bar{\Omega}) : v_{h|T} \in P_1(T)\ \forall T \in \mathcal{T}, v_{|\partial\Omega} = g\}.$$

Také výsledná aproximace úlohy (P) metodou konečných prvů je formálně stejná jako ve dvoudimenzionálním případě:

$$\left.\begin{array}{l} \text{Najdi } u_h \in V_{hg} \quad \text{tak, že} \\ \displaystyle\int_\Omega \nabla u_h \cdot \nabla v_h + \alpha u_h v_h = \int_\Omega f v_h \quad \forall v_h \in V_{h0}. \end{array}\right\} \ (P_h)$$

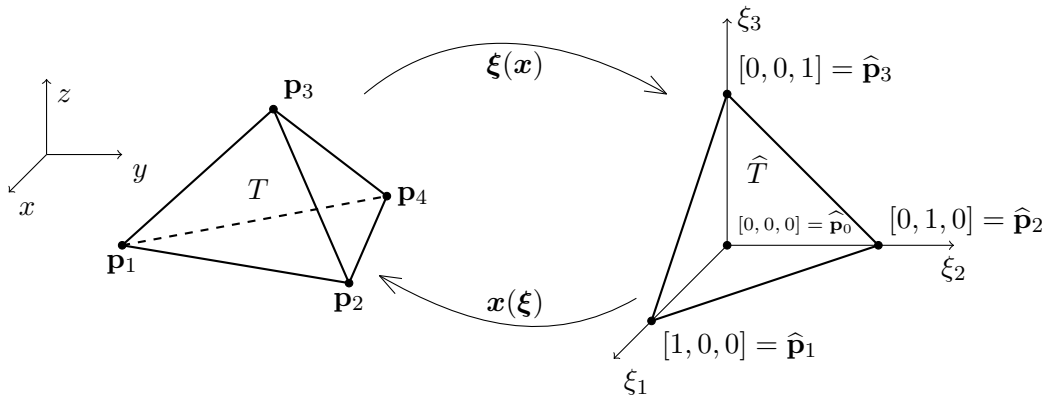
V MATLABu budeme datové struktury reprezentovat takto:

$$\texttt{p} = [x_1, y_1, z_1; x_2, y_2, z_2; \ldots]';\ \texttt{np} = \texttt{size}(\texttt{p}, 1);$$
$$\texttt{t} = [\ldots; i, j, k, l; \ldots];\ \texttt{nt} = \texttt{size}(\texttt{t}, 1);$$

kde i, j, k, l jsou indexy vrcholů čtyřstěnu T_{ijkl}.

4.2 Transformace na referenční čtyřstěn

Na uvažovaném čtyřstěnu $T = T_{ijkl}$ budeme používat lokální číslování uzlů: $\mathbf{p}_i = \mathbf{p}_1 = [x_1, y_1, z_1]^\mathsf{T}$ a podobně pro $\mathbf{p}_j = \mathbf{p}_2$, $\mathbf{p}_k = \mathbf{p}_3$ a $\mathbf{p}_l = \mathbf{p}_4$, viz obrázek 10.



Obrázek 10: Lokální číslování uzlů a referenční čtyřstěn

Na čtyřstěnu jsou čtyři nenulové lineární bázové funkce $\phi_1(\boldsymbol{x})$, $\phi_2(\boldsymbol{x})$, $\phi_3(\boldsymbol{x})$, $\phi_4(\boldsymbol{x}) \in P_1(T)$, $\boldsymbol{x} = [x, y, z]^\mathsf{T} \in T$. Explicitní předpis těchto bázových funkcí potřebovat nebudeme, protože obdobně jako v minulých kapitolách použijeme lineární transformaci, která bude tentokrát z čtyřstěnu T na referenční čtyřstěn $\widehat{T}$ s vrcholy $\widehat{\mathbf{p}}_0 = [0,0,0]^\mathsf{T}$, $\widehat{\mathbf{p}}_1 = [1,0,0]^\mathsf{T}$, $\widehat{\mathbf{p}}_2 = [0,1,0]^\mathsf{T}$, $\widehat{\mathbf{p}}_3 = [0,0,1]^\mathsf{T}$. Příslušné referenční bázové funkce mají předpis:

$$\begin{aligned}
\widehat{\phi}_0(\boldsymbol{\xi}) &= 1 - \xi_1 - \xi_2 - \xi_3, \\
\widehat{\phi}_1(\boldsymbol{\xi}) &= \xi_1, \\
\widehat{\phi}_2(\boldsymbol{\xi}) &= \xi_2, \\
\widehat{\phi}_3(\boldsymbol{\xi}) &= \xi_3, \qquad \boldsymbol{\xi} = [\xi_1, \xi_2, \xi_3]^\mathsf{T} \in \widehat{T}.
\end{aligned}$$

Obdobně jako ve 2D použijeme referenční bázové funkce jako *barycentrické souřadnice* bodu $\boldsymbol{x} = [x, y, z]^\mathsf{T} \in T$ vzhledem k vrcholům čtyřstěnu T:

$$\boldsymbol{x} = \mathbf{p}_1\widehat{\phi}_0(\boldsymbol{\xi}) + \mathbf{p}_2\widehat{\phi}_1(\boldsymbol{\xi}) + \mathbf{p}_3\widehat{\phi}_2(\boldsymbol{\xi}) + \mathbf{p}_4\widehat{\phi}_3(\boldsymbol{\xi}), \quad \boldsymbol{\xi} \in \widehat{T}.$$

Lineární transformace čtyřstěnu $\widehat{T}$ na čtyřstěn T je dána předpisem:

$$\boldsymbol{x} = \boldsymbol{x}(\boldsymbol{\xi}) : \widehat{T} \to T. \tag{16}$$

Můžeme ji zapsat také takto:

$$\boldsymbol{x} = \mathbf{p}_1 + \mathbf{X}\boldsymbol{\xi}, \quad \mathbf{X} = [\mathbf{p}_2 - \mathbf{p}_1,\ \mathbf{p}_3 - \mathbf{p}_1,\ \mathbf{p}_4 - \mathbf{p}_1] \in \mathbb{R}^{3\times 3}.$$

Formálně stejně jako ve 2D zapíšeme transformaci inverzní:

$$\boldsymbol{\xi} = \boldsymbol{\xi}(\boldsymbol{x}) : \ T \to \widehat{T},$$

přičemž

$$\boldsymbol{\xi} = \mathbf{X}^{-1}(\boldsymbol{x} - \mathbf{p}_1). \tag{17}$$

Složitější bude vyjádření inverze $\mathbf{X}^{-1}$, protože Cramerovo pravidlo musíme nyní aplikovat na matici třetího řádu. Pro zjednodušení zápisu označíme: $x_{ij} = x_i - x_j,\ y_{ij} = y_i - y_j,\ z_{ij} = z_i - z_j,\ i, j = 1, 2, 3, 4$. Matici $\mathbf{X}$ pak lze zapsat ve tvaru:

$$\mathbf{X} = \begin{bmatrix} x_{21} & x_{31} & x_{41} \\ y_{21} & y_{31} & y_{41} \\ z_{21} & z_{31} & z_{41} \end{bmatrix}$$

a její determinant lze vypočíst Sarrusovým pravidlem:

$$|\mathbf{X}| = x_{21}(y_{31}z_{41} - y_{41}z_{31}) + x_{31}(y_{41}z_{21} - y_{21}z_{41}) + x_{41}(y_{21}z_{31} - y_{31}z_{21}).$$

Dále je známo, že pro absolutní hodnotu determinantu matice $\mathbf{X}$ platí: $\mathrm{abs}(|\mathbf{X}|) = 6|T|$, kde $|T|$ je objem čtyřstěnu. Lze tedy říci, že determinant $|\mathbf{X}|$ je nenulový pro každý nedegradovaný čtyřstěn a matice $\mathbf{X}$ je v takovém případě invertovatelná. Inverzní matici $\mathbf{X}^{-1}$ zapíšeme ve tvaru:

$$\mathbf{X}^{-1} = \frac{1}{|\mathbf{X}|} \begin{bmatrix} \bar{x}_{11} & \bar{x}_{21} & \bar{x}_{31} \\ \bar{x}_{12} & \bar{x}_{22} & \bar{x}_{32} \\ \bar{x}_{13} & \bar{x}_{23} & \bar{x}_{33} \end{bmatrix}.$$

Prvky $\bar{x}_{ij},\ i, j = 1, 2, 3$ jsou určeny vzorci:

$$\begin{aligned}
&\bar{x}_{11} = y_{31}z_{41} - y_{41}z_{31}, && \bar{x}_{21} = y_{41}z_{21} - y_{21}z_{41}, && \bar{x}_{31} = y_{21}z_{31} - y_{31}z_{21}, \\
&\bar{x}_{12} = z_{31}x_{41} - z_{41}x_{31}, && \bar{x}_{22} = z_{41}x_{21} - z_{21}x_{41}, && \bar{x}_{32} = z_{21}x_{31} - z_{31}x_{21}, \\
&\bar{x}_{13} = x_{31}y_{41} - x_{41}y_{31}, && \bar{x}_{23} = x_{41}y_{21} - x_{21}y_{41}, && \bar{x}_{33} = x_{21}y_{31} - x_{31}y_{21}.
\end{aligned}$$

Později budeme potřebovat záporné sloupcové součty matice $\mathbf{X}^{-1}$, které vypočítáme vhodným použitím determinantu:

$$\bar{x}_{11} + \bar{x}_{21} + \bar{x}_{31} = \begin{vmatrix} 1 & 1 & 1 \\ y_{21} & y_{31} & y_{41} \\ z_{21} & z_{31} & z_{41} \end{vmatrix} = \begin{vmatrix} 1 & 1 & 1 \\ 0 & y_{32} & y_{42} \\ 0 & z_{32} & z_{42} \end{vmatrix} = y_{32}\, z_{42} - y_{42}\, z_{32},$$

$$\bar{x}_{12}+\bar{x}_{22}+\bar{x}_{32}=\begin{vmatrix} x_{21} & x_{31} & x_{41} \\ 1 & 1 & 1 \\ z_{21} & z_{31} & z_{41} \end{vmatrix}=\begin{vmatrix} 0 & x_{32} & x_{42} \\ 1 & 1 & 1 \\ 0 & z_{32} & z_{42} \end{vmatrix}=z_{32}\,x_{42}-z_{42}\,x_{32},$$

$$\bar{x}_{13}+\bar{x}_{23}+\bar{x}_{33}=\begin{vmatrix} x_{21} & x_{31} & x_{41} \\ y_{21} & y_{31} & y_{41} \\ 1 & 1 & 1 \end{vmatrix}=\begin{vmatrix} 0 & x_{32} & x_{42} \\ 0 & y_{32} & y_{42} \\ 1 & 1 & 1 \end{vmatrix}=x_{32}\,y_{42}-x_{42}\,y_{32}.$$

Výsledné záporné sloupcové součty mají tvar:

$$\begin{aligned} -(\bar{x}_{11}+\bar{x}_{21}+\bar{x}_{31}) &= y_{42}\,z_{32}-y_{32}\,z_{42}, \\ -(\bar{x}_{12}+\bar{x}_{22}+\bar{x}_{32}) &= x_{32}\,z_{42}-x_{42}\,z_{32}, \\ -(\bar{x}_{13}+\bar{x}_{23}+\bar{x}_{33}) &= x_{42}\,y_{32}-x_{32}\,y_{42}. \end{aligned}$$

4.3 Sestavení soustavy lineárních rovnic

Podobně jako v předchozích kapitolách sestavíme k úloze (P_h) nejprve soustavu lineárních rovnic bez okrajových podmínek:

$$\bar{\mathbf{A}}\bar{\boldsymbol{u}}=\bar{\boldsymbol{b}},\quad \bar{\mathbf{A}}\in\mathbb{R}^{n\times n},\quad \bar{\boldsymbol{b}}\in\mathbb{R}^{n},\quad \bar{\boldsymbol{u}}\in\mathbb{R}^{n},$$

kde $\bar{\mathbf{A}}$ je obdobně jako ve 2D *rozšířená matice tuhosti*, $\bar{\mathbf{R}}$ je *matice tuhosti* a $\bar{\mathbf{M}}$ je *matice hmotnosti*. Tedy

$$\bar{\mathbf{A}}=\bar{\mathbf{R}}+\bar{\mathbf{M}}.$$

Podobně budeme na čtyřstěnu $T=T_{ijkl}\in\mathcal{T}$ rozlišovat lokální matici tuhosti $\mathbf{R}_T\in\mathbb{R}^{4\times4}$ a lokální matici hmotnosti $\mathbf{M}_T\in\mathbb{R}^{4\times4}$, které společně tvoří rozšířenou lokální matici tuhosti:

$$\mathbf{A}_T=\mathbf{R}_T+\mathbf{M}_T.$$

Složky lokální matice hmotnosti $\mathbf{M}_T$ mají pro $i,j=1,2,3,4$ tvar:

$$(\mathbf{M}_T)_{ij}=\alpha\int_T\phi_j(\boldsymbol{x})\phi_i(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x}=\alpha\int_{\widehat{T}}\phi_j(\boldsymbol{x}(\boldsymbol{\xi}))\phi_i(\boldsymbol{x}(\boldsymbol{\xi}))\;|\mathbf{X}|\,\mathrm{d}\boldsymbol{\xi}=\alpha\,6|T|\int_{\widehat{T}}\widehat{\phi}_{j-1}(\boldsymbol{\xi})\widehat{\phi}_{i-1}(\boldsymbol{\xi})\,\mathrm{d}\boldsymbol{\xi}.$$

Po vypočtení všech šestnácti integrálu, viz dodatky B.2, pro referenční bázové funkce na referenčním čtyřstěnu získáme předpis lokální matice hmotnosti:

$$\mathbf{M}_T=\frac{\alpha|T|}{20}\begin{bmatrix} 2 & 1 & 1 & 1 \\ 1 & 2 & 1 & 1 \\ 1 & 1 & 2 & 1 \\ 1 & 1 & 1 & 2 \end{bmatrix}. \tag{18}$$

S využitím tohoto vztahu lze sestavení globální matice hmotnosti $\bar{\mathbf{M}}$ implementovat v MATLABu kódem 13. Toto sestavení lze stejně jako v minulých kapitolách zapsat vektorizovaně 14

```
M=sparse(np,np);
Mel=(1/20)*[2 1 1 1; 1 2 1 1; 1 1 2 1; 1 1 1 2];
% Element of mass matrix
for k=1:nt
   ind=t(k,:);
   x21=p(ind(2),1)-p(ind(1),1); x31=p(ind(3),1)-p(ind(1),1);
   x41=p(ind(4),1)-p(ind(1),1);
   y21=p(ind(2),2)-p(ind(1),2); z21=p(ind(2),3)-p(ind(1),3);
   y31=p(ind(3),2)-p(ind(1),2); z31=p(ind(3),3)-p(ind(1),3);
   y41=p(ind(4),2)-p(ind(1),2); z41=p(ind(4),3)-p(ind(1),3);
   y32=p(ind(3),2)-p(ind(2),2); z32=p(ind(3),3)-p(ind(2),3);
   y42=p(ind(4),2)-p(ind(2),2); z42=p(ind(4),3)-p(ind(2),3);
   xt=[z32*y42-y32*z42;y31*z41-z31*y41;z21*y41-y21*z41;y21*z31-z21*y31];
   tvolume=(x21*xt(2)+x31*xt(3)+x41*xt(4))/6;
   M(ind,ind)=M(ind,ind)+alpha*tvolume*Mel;
end
```

Kód 13: Sestavení globální matice hmotnosti

```
M=sparse(np,np);
% Tetrahedron volume
x21=p(t(:,2),1)-p(t(:,1),1); x31=p(t(:,3),1)-p(t(:,1),1);
x41=p(t(:,4),1)-p(t(:,1),1);
y21=p(t(:,2),2)-p(t(:,1),2); z21=p(t(:,2),3)-p(t(:,1),3);
y31=p(t(:,3),2)-p(t(:,1),2); z31=p(t(:,3),3)-p(t(:,1),3);
y41=p(t(:,4),2)-p(t(:,1),2); z41=p(t(:,4),3)-p(t(:,1),3);
y32=p(t(:,3),2)-p(t(:,2),2); z32=p(t(:,3),3)-p(t(:,2),3);
y42=p(t(:,4),2)-p(t(:,2),2); z42=p(t(:,4),3)-p(t(:,2),3);
xt=[z32.*y42-y32.*z42,y31.*z41-z31.*y41,z21.*y41-y21.*z41,y21.*z31-z21.*y31];
tvolume=(x21.*xt(:,2)+x31.*xt(:,3)+x41.*xt(:,4))/6; aux=alpha*tvolume/20;
for i=1:4
    for j=1:i
      M=M+sparse(t(:,i),t(:,j),aux,np,np)+sparse(t(:,j),t(:,i),aux,np,np);
    end
end
```

Kód 14: Vektorizované sestavení matice $\mathbf{M}$

Lokální matici tuhosti $\bar{\mathbf{R}}_T$ tvoří složky:

$$(\mathbf{R}_T)_{ij} = \int_T \nabla\phi_j(\boldsymbol{x}) \cdot \nabla\phi_i(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x}, \quad i,j = 1,2,3,4.$$

Znovu využijeme substituci na referenční čtyřstěn $\widehat{T}$, přičemž z předchozí kapitoly již známe vztah mezi bázovou a referenční bázovou funkcí:

$$\nabla\phi_i(\boldsymbol{x}) = \nabla\widehat{\phi}_{i-1}(\boldsymbol{\xi})\mathbf{X}^{-1}, \quad i = 1,2,3,4. \tag{19}$$

Gradienty referenčních bázových funkcí snadno dopočítáme:

$$\nabla\widehat{\phi}_0 = [-1,-1,-1], \quad \nabla\widehat{\phi}_1 = [1,0,0], \quad \nabla\widehat{\phi}_2 = [0,1,0], \quad \nabla\widehat{\phi}_3 = [0,0,1].$$

Gradienty bázových funkcí na T mají s využitím vztahu (19) a inverzní matice $\mathbf{X}^{-1}$ tvar:

$$\begin{aligned}
\nabla\phi_1(\boldsymbol{x}) &= (|\mathbf{X}|)^{-1}[y_{42}\,z_{32} - y_{32}\,z_{42}, x_{32}\,z_{42} - x_{42}\,z_{32}, x_{42}\,y_{32} - x_{32}\,y_{42}],\\
\nabla\phi_2(\boldsymbol{x}) &= (|\mathbf{X}|)^{-1}[y_{31}\,z_{41} - y_{41}\,z_{31}, z_{31}\,x_{41} - z_{41}\,x_{31}, x_{31}\,y_{41} - x_{41}\,y_{31}],\\
\nabla\phi_3(\boldsymbol{x}) &= (|\mathbf{X}|)^{-1}[y_{41}\,z_{21} - y_{21}\,z_{41}, z_{41}\,x_{21} - z_{21}\,x_{41}, x_{41}\,y_{21} - x_{21}\,y_{41}],\\
\nabla\phi_4(\boldsymbol{x}) &= (|\mathbf{X}|)^{-1}[y_{21}\,z_{31} - y_{31}\,z_{21}, z_{21}\,x_{31} - z_{31}\,x_{21}, x_{21}\,y_{31} - x_{31}\,y_{21}].
\end{aligned}$$

Pro zjednodušení zavedeme označení:

$$\boldsymbol{x}_T = \begin{bmatrix} y_{42}\,z_{32} - y_{32}\,z_{42} \\ y_{31}\,z_{41} - y_{41}\,z_{31} \\ y_{41}\,z_{21} - y_{21}\,z_{41} \\ y_{21}\,z_{31} - y_{31}\,z_{21} \end{bmatrix}, \quad \mathbf{y}_T = \begin{bmatrix} x_{32}\,z_{42} - x_{42}\,z_{32} \\ z_{31}\,x_{41} - z_{41}\,x_{31} \\ z_{41}\,x_{21} - z_{21}\,x_{41} \\ z_{21}\,x_{31} - z_{31}\,x_{21} \end{bmatrix}, \quad \mathbf{z}_T = \begin{bmatrix} x_{42}\,y_{32} - x_{32}\,y_{42} \\ x_{31}\,y_{41} - x_{41}\,y_{31} \\ x_{41}\,y_{21} - x_{21}\,y_{41} \\ x_{21}\,y_{31} - x_{31}\,y_{21} \end{bmatrix}.$$

Pak platí:

$$\left[\frac{\partial\phi_1}{\partial x}, \frac{\partial\phi_2}{\partial x}, \frac{\partial\phi_3}{\partial x}, \frac{\partial\phi_4}{\partial x}\right]^{\mathsf{T}} = \frac{1}{|\mathbf{X}|}\boldsymbol{x}_T, \quad \left[\frac{\partial\phi_1}{\partial y}, \frac{\partial\phi_2}{\partial y}, \frac{\partial\phi_3}{\partial y}, \frac{\partial\phi_4}{\partial y}\right]^{\mathsf{T}} = \frac{1}{|\mathbf{X}|}\mathbf{y}_T,$$

$$\left[\frac{\partial\phi_1}{\partial z}, \frac{\partial\phi_2}{\partial z}, \frac{\partial\phi_3}{\partial z}, \frac{\partial\phi_4}{\partial z}\right]^{\mathsf{T}} = \frac{1}{|\mathbf{X}|}\mathbf{z}_T.$$

Výslednou matici $\mathbf{R}_T$ lze zapsat následovně, protože absolutní hodnota $|\mathbf{X}|$ je $6|T|$:

$$\mathbf{R}_T = \frac{1}{36|T|}(\boldsymbol{x}_T\,\boldsymbol{x}_T^{\mathsf{T}} + \mathbf{y}_T\,\mathbf{y}_T^{\mathsf{T}} + \mathbf{z}_T\,\mathbf{z}_T^{\mathsf{T}}). \tag{20}$$

Kód v MATLABu pro sestavení globální matice tuhosti za pomocí cyklů může být, jako v kódu 15.

```
R=sparse(np,np);
for k=1:nt
    ind=t(k,:);
    x21=p(ind(2),1)-p(ind(1),1); y21=p(ind(2),2)-p(ind(1),2);
    x31=p(ind(3),1)-p(ind(1),1); y31=p(ind(3),2)-p(ind(1),2);
    x41=p(ind(4),1)-p(ind(1),1); y41=p(ind(4),2)-p(ind(1),2);
    x32=p(ind(3),1)-p(ind(2),1); y32=p(ind(3),2)-p(ind(2),2);
    x42=p(ind(4),1)-p(ind(2),1); y42=p(ind(4),2)-p(ind(2),2);
    z21=p(ind(2),3)-p(ind(1),3); z31=p(ind(3),3)-p(ind(1),3);
    z41=p(ind(4),3)-p(ind(1),3); z32=p(ind(3),3)-p(ind(2),3);
    z42=p(ind(4),3)-p(ind(2),3);
    xt=[z32*y42-y32*z42;y31*z41-z31*y41;z21*y41-y21*z41;y21*z31-z21*y31];
    yt=[x32*z42-z32*x42;z31*x41-x31*z41;x21*z41-z21*x41;z21*x31-x21*z31];
    zt=[y32*x42-x32*y42;x31*y41-y31*x41;y21*x41-x21*y41;x21*y31-y21*x31];
    tvolume=(x21*xt(2)+x31*xt(3)+x41*xt(4))/6;
    % Stiffness matrix increment
    R(ind,ind)=R(ind,ind)+(1/36/tvolume)*(xt*xt'+yt*yt'+zt*zt');
end
```

Kód 15: Standardní sestavení matice tuhosti

Tento kód lze samozřejmě vektorizovat, což je zapsáno v kódu 16.

```
R=sparse(np,np);
x21=p(t(:,2),1)-p(t(:,1),1); y21=p(t(:,2),2)-p(t(:,1),2);
x31=p(t(:,3),1)-p(t(:,1),1); y31=p(t(:,3),2)-p(t(:,1),2);
x41=p(t(:,4),1)-p(t(:,1),1); y41=p(t(:,4),2)-p(t(:,1),2);
x32=p(t(:,3),1)-p(t(:,2),1); y32=p(t(:,3),2)-p(t(:,2),2);
x42=p(t(:,4),1)-p(t(:,2),1); y42=p(t(:,4),2)-p(t(:,2),2);
z21=p(t(:,2),3)-p(t(:,1),3); z31=p(t(:,3),3)-p(t(:,1),3);
z41=p(t(:,4),3)-p(t(:,1),3); z32=p(t(:,3),3)-p(t(:,2),3);
z42=p(t(:,4),3)-p(t(:,2),3);
xt=[z32.*y42-y32.*z42,y31.*z41-z31.*y41,z21.*y41-y21.*z41,y21.*z31-z21.*y31];
yt=[x32.*z42-z32.*x42,z31.*x41-x31.*z41,x21.*z41-z21.*x41,z21.*x31-x21.*z31];
zt=[y32.*x42-x32.*y42,x31.*y41-y31.*x41,y21.*x41-x21.*y41,x21.*y31-y21.*x31];
tvolume=(x21.*xt(:,2)+x31.*xt(:,3)+x41.*xt(:,4))/6;tau=(1/36)/tvolume;
```

```
% Stiffness matrix increment
for i=1:4
    for j=1:4
      R=R+sparse(t(:,i),t(:,j),tau*(xt(:,i).*xt(:,j)+yt(:,i).*yt(:,j)...
                                                    +zt(:,i).*zt(:,j)),np,np);
    end
end
```

Kód 16: Vektorizované sestavení matice tuhosti

Zbývá ukázat sestavení lokálního vektoru pravé strany $\bar{\boldsymbol{b}} \in \mathbb{R}^4$, jehož jednotlivé prvky mají tvar:

$$(\boldsymbol{b}_T)_i = \int_T f\phi_i \,\mathrm{d}\boldsymbol{x}, \quad i = 1,2,3,4.$$

S použitím numerické integrace nahradíme funkci f na T průměrem funkčních hodnot z vrcholových bodů čtyřstěnu T, tj.

$$f_T = \frac{1}{4}(f(\mathbf{p}_1) + f(\mathbf{p}_2) + f(\mathbf{p}_3) + f(\mathbf{p}_4)),$$

takže dostáváme

$$(\boldsymbol{b}_T)_i \doteq f_T \int_T \phi_i \,\mathrm{d}\boldsymbol{x} \doteq 6|T| f_T \int_{\widehat{T}} \widehat{\phi}_{i-1} \,\mathrm{d}\boldsymbol{\xi}, \quad i = 1,2,3,4.$$

Jelikož $\int_{\widehat{T}} \widehat{\phi}_{i-1} \,\mathrm{d}\boldsymbol{\xi} = \frac{1}{24}$, $i = 1,2,3,4$, pak je zřejmé, že

$$\boldsymbol{b}_T = \frac{f_T|T|}{4}\,[1,1,1,1]^\mathsf{T}. \tag{21}$$

V MATLABu uvedeme nejprve nevektorizovaný kód sestavení:

```
b=zeros(np,1);
fp=f(p(:,1),p(:,2),p(:,3));
for k=1:nt
   ind=t(k,:);
   ft=(fp(ind(1))+fp(ind(2))+fp(ind(3))+fp(ind(4)))/4;
   b(ind)=b(ind)+ft*tvolume/4*[1;1;1;1];
end
```

Kód 17: Sestavení vektoru pravé strany

A dále vektorizovaná verze sestavení:

```
b=zeros(np,1);
ft=(fp(t(:,1))+fp(t(:,2))+fp(t(:,3))+fp(t(:,4)))/4;
tvolume=(x21.*xt(:,2)+x31.*xt(:,3)+x41.*xt(:,4))/6;
bux=(1/4)*ft.*tvolume;
for i=1:4
   b=b+sparse(t(:,i),1,bux,np,1);
end
```

Kód 18: Vektorizované sestavení vektoru pravé strany

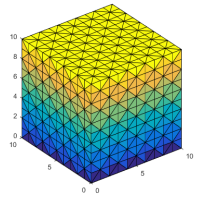
Popis opět zakončíme realizací okrajových podmínek, kde budeme předpokládat, že proměnná `dind` obsahuje indexy všech hraničních uzlů. Následující kód je totožný s 2D případem, jen výpočet proměnné `d` je rozšířen do tří dimenzí.

```
% Matrix from a stiffness and mass matrices
A=R+M;
% Boundary conditions
d=g(p(dind,1),p(dind,2),p(dind,3));
A(dind,:)=[]; b(dind)=[];
b=b-A(:,dind)*d;
A(:,dind)=[];
% Solve the linear system
u=A\b;
ind_all=(1:np)';
ind_minus=setdiff(ind_all,dind);
uu(ind_minus)=u;
% Problem solution
uu(dind)=d;
```

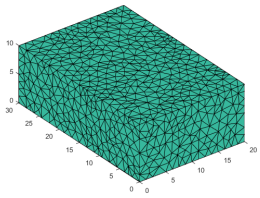
Kód 19: Realizace okrajových podmínek a vyřešení lineární soustavy

4.4 Výpočetní experimenty

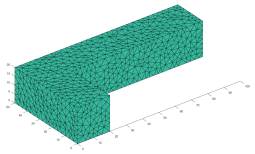
Obdobně jako v předchozím 1D a 2D případě byly provedeny experimenty s časovou náročností jednotlivých implementací. Nejprve byl proveden výpočet pro krychli o rozměrech $1 \times 1 \times 1$ s rovnoměrně rozloženou sítí, viz obrázek 11. Dále pak kvádr, L-tvar a válec, viz obrázky 12-14. Výsledné časy vektorizovaného sestavení jsou i v případě 3D případu velmi nízké v porovnání se standardním sestavováním po jednotlivých elementech, viz tabulky 11-14. Sítě byly vytvořeny pomocí balíku ISO2MESH, viz dodatek C.1.



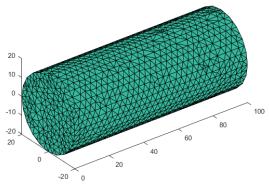
Obrázek 11: Krychle



Obrázek 12: Kvádr



Obrázek 13: L-tvar



Obrázek 14: Válec

Tabulka 5: Výsledky časových experimentů pro krychli

Počet elementů	1080	2560	8640	20480	53240	163840	486680
Standardní sestavení (s)	0.0745	0.1838	0.7934	2.5561	14.96	186.6	2451.8
Vektorizované sestavení (s)	0.0053	0.0119	0.0441	0.1054	0.3667	1.3697	4.3961
Poměr	7.092%	6.501%	5.6%	4.122%	2.451%	0.734%	0.1793%

Tabulka 6: Výsledky časových experimentů pro kvádr

Počet elementů	1061	2484	8560	20655	53395	164049	486124
Standardní sestavení (s)	0.0739	0.1858	0.7730	3.9047	24.29	272.2	2645.5
Vektorizované sestavení (s)	0.0055	0.0127	0.0481	0.1183	0.4103	1.5012	4.7955
Poměr	7.465%	6.811%	6.2%	3.03%	1.689%	0.552%	0.1813%

Tabulka 7: Výsledky časových experimentů pro L-tvar

Počet elementů	1098	2536	8626	20412	53163	163597	486693
Standardní sestavení (s)	0.0766	0.2052	0.7756	3.7861	23.58	269.6	2588.1
Vektorizované sestavení (s)	0.0057	0.0124	0.0487	0.1167	0.4065	1.5020	4.8399
Poměr	7.421%	6.1%	6.277%	3.081%	1.724%	0.557%	0.187%

Tabulka 8: Výsledky časových experimentů pro válec

Počet elementů	1081	2557	8567	20663	54263	163040	489675
Standardní sestavení (s)	0.0746	0.1828	0.7618	3.7253	24.27	256.5	2573.8
Vektorizované sestavení (s)	0.0056	0.0123	0.0484	0.1178	0.4207	1.5002	4.8805
Poměr	7.501%	6.7%	6.358%	3.162%	1.733%	0.585%	0.189%

4.5 Bublinkové funkce ve 3D

Nyní rozšíříme aproximaci úlohy (P) odvozené z modelové úlohy (15) ze začátku kapitoly 4 o bublinkové funkce obdobně jako v sekci 3.5. Prostor bublinkových funkcí zavedeme následovně:

$$B_h = \{v_h \in C(\bar{\Omega}) : v_{h|T} = k_T\, \phi_{Tb} \quad \forall T \in \mathcal{T},\ k_t \in \mathbb{R}\}.$$

Bublinková funkce je pak na každém dělení $T \in \mathcal{T}$ definována takto:

$$\phi_b(\boldsymbol{x}) = 4^4\, \phi_1(\boldsymbol{x})\phi_2(\boldsymbol{x})\phi_3(\boldsymbol{x})\phi_4(\boldsymbol{x}), \quad \boldsymbol{x} \in T.$$

Výsledná aproximace (P_h^b) úlohy (P) má formálně stejný tvar jako v případě 2D:

$$\left.\begin{array}{l} \text{Najdi } u_h \in V_{hg} \oplus B_h \text{ tak, že} \\ \displaystyle\int_\Omega \nabla u_h \cdot \nabla v_h + \alpha u_h v_h = \int_\Omega f v_h \quad \forall v_h \in V_{h0} \oplus B_h. \end{array}\right\} (P_h^b).$$

Nadále platí lemma z kapitoly 3.5 rozšířené o další dimenzi, tj.

Lemma 2

$$\nabla\phi_1(\boldsymbol{x}) + \nabla\phi_2(\boldsymbol{x}) + \nabla\phi_3(\boldsymbol{x}) + \nabla\phi_4(\boldsymbol{x}) = 0, \quad \forall \boldsymbol{x} \in T.$$

Na $\widehat{T}$ přibyde referenční bublinková funkce:

$$\widehat{\phi}_b(\boldsymbol{\xi}) = 4^4\, \widehat{\phi}_0(\boldsymbol{\xi})\widehat{\phi}_1(\boldsymbol{\xi})\widehat{\phi}_2(\boldsymbol{\xi})\widehat{\phi}_3(\boldsymbol{\xi}), \quad \boldsymbol{\xi} \in \widehat{T}.$$

4.6 Sestavení soustavy s bublinkovými funkcemi

Nyní sestavíme k úloze (P_h^b) odpovídající soustavu lineárních rovnic opět nejprve bez Dirichletových okrajových podmínek:

$$\bar{\mathbf{A}}\bar{\boldsymbol{u}} = \bar{\boldsymbol{b}},$$

kde $\bar{\mathbf{A}} \in \mathbb{R}^{n\times n}$, $\bar{\boldsymbol{b}} \in \mathbb{R}^n$ a $\bar{\boldsymbol{u}} \in \mathbb{R}^n$. Začneme odvozením matic $\mathbf{R}_T^b$ a $\mathbf{M}_T^b$, které jsou složkami matice $\bar{\mathbf{A}}$. Veškeré integrály byly počítány ručně nebo s použitím toolboxu SYMBOLIC v MATLABu, viz B.2.

Budeme postupovat obdobně jako v kapitole 3.6. Lokální *matice hmotnosti* má tvar:

$$\mathbf{M}_T^b = \begin{bmatrix} \mathbf{M}_T & \boldsymbol{m}_T \\ \boldsymbol{m}_T^\mathsf{T} & \omega_M \end{bmatrix} \in \mathbb{R}^{5\times 5},$$

kde matice $\mathbf{M}_T \in \mathbb{R}^{4\times 4}$ je dána vztahem (18). Dále víme, že pro složky vektoru $\boldsymbol{m}_T \in \mathbb{R}^4$ platí:

$$(\boldsymbol{m}_T)_i = \alpha \int_T \phi_b\phi_i, \qquad i = 1,2,3,4.$$

Použijeme-li substituci na referenční čtyřstěn $\widehat{T}$, na kterém je číslování uzlů invariantní vůči přečíslení, získáme následující rovnost:

$$(\boldsymbol{m}_T)_1 = (\boldsymbol{m}_T)_2 = (\boldsymbol{m}_T)_3 = (\boldsymbol{m}_T)_4.$$

Stačí tedy vypočítat prvek $(\boldsymbol{m}_T)_1$:

$$(\boldsymbol{m}_T)_1 = \alpha \int_T \phi_b \phi_1 = 6 \cdot 4^4 \alpha |T| \int_{\widehat{T}} \widehat{\phi}_0(\boldsymbol{\xi})^2 \widehat{\phi}_1(\boldsymbol{\xi}) \widehat{\phi}_2(\boldsymbol{\xi}) \widehat{\phi}_3(\boldsymbol{\xi}) \, \mathrm{d}\boldsymbol{\xi} = \alpha |T| \frac{8}{105}.$$

Výsledný vektor $\boldsymbol{m}_T$ pak vypadá takto:

$$\boldsymbol{m}_T = \frac{8\alpha|T|}{105} [1,1,1,1]^\mathsf{T}.$$

Pro $\omega_M \in \mathbb{R}$ platí:

$$\omega_M = \int_T \phi_b(\boldsymbol{x})^2 \, \mathrm{d}\boldsymbol{x} = 6\alpha|T| \int_{\widehat{T}} \widehat{\phi}_b(\boldsymbol{\xi})^2 \, \mathrm{d}\boldsymbol{\xi} = 6 \cdot 4^8 \alpha |T| \int_{\widehat{T}} \widehat{\phi}_0(\boldsymbol{\xi})^2 \widehat{\phi}_1(\boldsymbol{\xi})^2 \widehat{\phi}_2(\boldsymbol{\xi})^2 \widehat{\phi}_3(\boldsymbol{\xi})^2 \, \mathrm{d}\boldsymbol{\xi}$$

a po vyčíslení je

$$\omega_M = \alpha |T| \frac{621}{3940}.$$

Pro lokální *matici tuhosti* platí:

$$\mathbf{R}_T^b = \begin{bmatrix} \mathbf{R}_T & \boldsymbol{r}_T \\ \boldsymbol{r}_T^\mathsf{T} & \omega_R \end{bmatrix} \in \mathbb{R}^{5\times 5},$$

kde matice $\mathbf{R}_T$ je určena vztahem (20). Dále pro složky vektoru $\boldsymbol{r}_T \in \mathbb{R}^4$ platí:

$$(\boldsymbol{r}_T)_i = \int_T \nabla \phi_b \cdot \nabla \phi_i, \qquad i = 1,2,3,4,$$

Obdobným způsobem jako ve 2D určíme například hodnotu prvku $(\boldsymbol{r}_T)_1$, zde již zkráceným zápisem:

$$\begin{aligned}(\boldsymbol{r}_T)_1 = 4^4 \int_T \nabla(\phi_1\phi_2\phi_3\phi_4) \cdot \nabla\phi_1 = 4^4 c_1 (\nabla\phi_1 \cdot \nabla\phi_1 + \nabla\phi_2 \cdot \nabla\phi_1 + \nabla\phi_3 \cdot \nabla\phi_1 + \nabla\phi_4 \cdot \nabla\phi_1) = \\ = 4^4 c_1 (\nabla\phi_1 + \nabla\phi_2 + \nabla\phi_3 + \nabla\phi_4) \cdot \nabla\phi_1 = 0,\end{aligned}$$

což plyne z konstantních skalárních součinů gradientů a také proto, že pro dílčí integrály platí:

$$c_1 = \int_T \phi_2\phi_3\phi_4 = \int_T \phi_1\phi_3\phi_4 = \int_T \phi_1\phi_2\phi_4 = \int_T \phi_1\phi_2\phi_3 = \frac{|T|}{120}.$$

S pomocí našeho Lemmatu 2 jsme určili, že obdobně jako ve 2D případě je $\boldsymbol{r}_T = \mathbf{0}$. Pomocí podobných úvah dostaneme:

$$\begin{aligned}\omega_R = \int_T \nabla\phi_b \cdot \nabla\phi_b = 4^8 c_2 (\quad & \nabla\phi_1 \cdot \nabla\phi_1 + \nabla\phi_2 \cdot \nabla\phi_2 + \nabla\phi_3 \cdot \nabla\phi_3 + \nabla\phi_4 \cdot \nabla\phi_4 + \\ & \nabla\phi_1 \cdot \nabla\phi_2 + \nabla\phi_1 \cdot \nabla\phi_3 + \nabla\phi_1 \cdot \nabla\phi_4 + \\ & \nabla\phi_2 \cdot \nabla\phi_3 + \nabla\phi_2 \cdot \nabla\phi_4 + \\ & \nabla\phi_3 \cdot \nabla\phi_4),\end{aligned}$$

protože

$$\begin{aligned} c_2 &= \int_T \phi_2^2\phi_3^2\phi_4^2 = \int_T \phi_1^2\phi_3^2\phi_4^2 = \int_T \phi_1^2\phi_2^2\phi_4^2 = \int_T \phi_1^2\phi_2^2\phi_3^2 = \\ &= 2\int_T \phi_1\phi_2\phi_3^2\phi_4^2 = 2\int_T \phi_1\phi_2^2\phi_3\phi_4^2 = 2\int_T \phi_1\phi_2^2\phi_3^2\phi_4 = \\ &= 2\int_T \phi_1^2\phi_2\phi_3\phi_4^2 = 2\int_T \phi_1^2\phi_2\phi_3^2\phi_4 = 2\int_T \phi_1^2\phi_2^2\phi_3\phi_4 = \frac{|T|}{7560}. \end{aligned}$$

Poslední součiny na každém řádku jsou po úpravě rovny nule díky našemu Lemmatu 2.
Dále využijeme lemma k dosazení za $\nabla\phi_2 \cdot \nabla\phi_3 = -\nabla\phi_1 \cdot \nabla\phi_3 - \nabla\phi_3 \cdot \nabla\phi_3 - \nabla\phi_4 \cdot \nabla\phi_3$ a pak $\nabla\phi_1 \cdot \nabla\phi_2 + \nabla\phi_2 \cdot \nabla\phi_2 = -\nabla\phi_3 \cdot \nabla\phi_2 - \nabla\phi_4 \cdot \nabla\phi_2$. Tímto dostaneme:

$$\begin{aligned} \omega_R &= 4^8 c_2(\nabla\phi_1 \cdot \nabla\phi_1 + \nabla\phi_2 \cdot \nabla\phi_2 + \nabla\phi_1 \cdot \nabla\phi_2 - \nabla\phi_4 \cdot \nabla\phi_3) = \\ &= 4^8 c_2(\nabla\phi_1 \cdot \nabla\phi_1 - \nabla\phi_3 \cdot \nabla\phi_2 - \nabla\phi_4 \cdot \nabla\phi_2 - \nabla\phi_4 \cdot \nabla\phi_3) \end{aligned}$$

a konečně

$$\begin{aligned} \omega_R = \frac{2048}{8505|T|}(x_{T1}^2 + y_{T1}^2 + z_{T1}^2 - x_{T2}x_{T3} - y_{T2}y_{T3} - z_{T2}z_{T3} - x_{T2}x_{T4} - \\ - y_{T2}y_{T4} - z_{T2}z_{T4} - x_{T3}x_{T4} - y_{T3}y_{T4} - z_{T3}z_{T4}). \end{aligned}$$

Jako poslední si ukážeme, jak vypadá lokální vektor pravé strany $\boldsymbol{b}_T^b \in \mathbb{R}^5$:

$$\boldsymbol{b}_T^b = \begin{bmatrix} \boldsymbol{b}_T \\ b_b \end{bmatrix},$$

kde vektor $\boldsymbol{b}_T \in \mathbb{R}^4$ je popsán vztahem (21) a b_b určíme následovně:

$$b_b = \int_T f\phi_b \,\mathrm{d}\boldsymbol{x} = 6|T|f_T \int_{\widehat{T}} \widehat{\phi}_b \,\mathrm{d}\boldsymbol{\xi} = 6|T|f_T 4^4 \int_{\widehat{T}} \widehat{\phi}_0\widehat{\phi}_1\widehat{\phi}_2\widehat{\phi}_3 \,\mathrm{d}\boldsymbol{\xi} = \frac{32}{105}|T|f_T.$$

Na lokální úrovni, tj. na čtyřstěnu T, uvažujme podobnou soustavu lineárních rovnic a stejný formální popis eliminace bublinkových složek jako u 2D úlohy.

Byly provedeny experimenty pro zjištění rozdílu časové náročnosti sestavení s a bez bublinkových funkcí. Výsledky jsou v tabulce 9. Z experimentů vyplývá, že sestavení s bublinkovými funkcemi je asi o 4-8 % výpočetně náročnější. Poslední hodnota 399 vteřin je jen orientační, zdůvodu omezení výpočtu nedostatečnou kapacitou paměti RAM, které citelně zpomaluje výpočet.

Tabulka 9: Časy pro vektorové sestavení větších soustav

Počet elementů	486680	1310720	3645000	10485760	30142840
Sestavení bez bubl (sec.)	4.4616	12.895	37.448	113.72	340.39
Sestavení s bubl (sec.)	4.7908	13.789	39.777	121.99	(399.41)

Následují kódy pro standardní sestavení (kód 20) a pro vektorizované sestavení (kód 21):

```
A=sparse(np,np); b=zeros(np,1);
Mel=(1/20)*[2 1 1 1; 1 2 1 1; 1 1 2 1; 1 1 1 2]; % element mass matrix
% Loop over all tetrahedrons
for k=1:nt
    ind=t(k,:);
    x21=p(ind(2),1)-p(ind(1),1); y21=p(ind(2),2)-p(ind(1),2);
    x31=p(ind(3),1)-p(ind(1),1); y31=p(ind(3),2)-p(ind(1),2);
    x41=p(ind(4),1)-p(ind(1),1); y41=p(ind(4),2)-p(ind(1),2);
    x32=p(ind(3),1)-p(ind(2),1); y32=p(ind(3),2)-p(ind(2),2);
    x42=p(ind(4),1)-p(ind(2),1); y42=p(ind(4),2)-p(ind(2),2);
    z21=p(ind(2),3)-p(ind(1),3); z32=p(ind(3),3)-p(ind(2),3);
    z31=p(ind(3),3)-p(ind(1),3); z42=p(ind(4),3)-p(ind(2),3);
    z41=p(ind(4),3)-p(ind(1),3);
    xt=[z32*y42-y32*z42;y31*z41-z31*y41;z21*y41-y21*z41;y21*z31-z21*y31];
    yt=[x32*z42-z32*x42;z31*x41-x31*z41;x21*z41-z21*x41;z21*x31-x21*z31];
    zt=[y32*x42-x32*y42;x31*y41-y31*x41;y21*x41-x21*y41;x21*y31-y21*x31];
    xtt=[-xt(2)*xt(3)-yt(2)*yt(3)-zt(2)*zt(3),
          -xt(2)*xt(4)-yt(2)*yt(4)-zt(2)*zt(4),
          -xt(3)*xt(4)-yt(3)*yt(4)-zt(3)*zt(4)];
    tvolume=abs(x21*xt(2)+x31*xt(3)+x41*xt(4))/6;
    ft=(fp(ind(1))+fp(ind(2))+fp(ind(3))+fp(ind(4)))/4;
    mt=8*alpha*tvolume/105*[1;1;1;1];
    omega=2048/8505*(xt(1)^2+yt(1)^2+zt(1)^2+sum(xtt))/tvolume...
                                                  +621*alpha*tvolume/3940;
    % Element matrix and vector with P1-bubble
    A(ind,ind)=A(ind,ind)+alpha*tvolume*Mel+(1/36/tvolume)...
              *(xt*xt'+yt*yt'+zt*zt')-1/omega*mt*mt';
    b(ind)=b(ind)+ft*tvolume*[1;1;1;1]/4-32/105*tvolume*ft/omega*mt;
end
```

Kód 20: Sestavení matice $\bar{\mathbf{A}}$ a vektoru $\bar{\boldsymbol{b}}$ s použitím cyklů

```
A=sparse(np,np); b=zeros(np,1);
x21=p(t(:,2),1)-p(t(:,1),1); y21=p(t(:,2),2)-p(t(:,1),2);
x31=p(t(:,3),1)-p(t(:,1),1); y31=p(t(:,3),2)-p(t(:,1),2);
x41=p(t(:,4),1)-p(t(:,1),1); y41=p(t(:,4),2)-p(t(:,1),2);
x32=p(t(:,3),1)-p(t(:,2),1); y32=p(t(:,3),2)-p(t(:,2),2);
x42=p(t(:,4),1)-p(t(:,2),1); y42=p(t(:,4),2)-p(t(:,2),2);
z21=p(t(:,2),3)-p(t(:,1),3); z32=p(t(:,3),3)-p(t(:,2),3);
z31=p(t(:,3),3)-p(t(:,1),3); z42=p(t(:,4),3)-p(t(:,2),3);
z41=p(t(:,4),3)-p(t(:,1),3);
xt=[z32.*y42-y32.*z42,y31.*z41-z31.*y41,z21.*y41-y21.*z41,y21.*z31-z21.*y31];
yt=[x32.*z42-z32.*x42,z31.*x41-x31.*z41,x21.*z41-z21.*x41,z21.*x31-x21.*z31];
zt=[y32.*x42-x32.*y42,x31.*y41-y31.*x41,y21.*x41-x21.*y41,x21.*y31-y21.*x31];
xtx=xt(:,2).*xt(:,3)+xt(:,2).*xt(:,4)+xt(:,3).*xt(:,4);
xty=yt(:,2).*yt(:,3)+yt(:,2).*yt(:,4)+yt(:,3).*yt(:,4);
xtz=zt(:,2).*zt(:,3)+zt(:,2).*zt(:,4)+zt(:,3).*zt(:,4);
xtt=xtx+xty+xtz;
tvolume=abs((x21.*xt(:,2)+x31.*xt(:,3)+x41.*xt(:,4))/6);
ft=(fp(t(:,1))+fp(t(:,2))+fp(t(:,3))+fp(t(:,4)))/4;
omega=2048/8505*(xt(:,1).^2+yt(:,1).^2+zt(:,1).^2-xtt)./tvolume...
                                                    +621*alpha*tvolume/3940;
aux=alpha*tvolume/20; mt=8*alpha*tvolume/105; tau=(1/36)./tvolume;
bux=(1/4)*ft.*tvolume-32/105*ft.*tvolume.*mt./omega;
for i=1:4
   % Mass matrix increment
    for j=1:i
        A=A+sparse(t(:,i),t(:,j),aux,np,np)...
           +sparse(t(:,j),t(:,i),aux,np,np);
    end
    % Stiffness matrix increment
    for j=1:4
        A=A+sparse(t(:,i),t(:,j),tau.*(xt(:,i).*xt(:,j)+yt(:,i).*yt(:,j)...
            +zt(:,i).*zt(:,j))-mt.*mt./omega,np,np);
    end
    % Right side vector increment
    b=b+sparse(t(:,i),1,bux,np,1);
end
```

Kód 21: Vektorizované sestavení matice $\bar{\mathbf{A}}$ a vektoru $\bar{\boldsymbol{b}}$

5 Vektorové úlohy ve 2D a 3D

Budeme se zabývat úlohou, která popisuje Stokesův model proudění tekutiny ve dvou (2D) a ve třech (3D) prostorových dimenzích.

5.1 Klasická formulace úloh

Nechť $\Omega \subset \mathbb{R}^d$, $d = 2, 3$, je omezená oblast s dostatečně hladkou hranicí $\partial\Omega$. Budeme hledat vektorovou funkci $\boldsymbol{u} : \bar{\Omega} \to \mathbb{R}^d$ a skalární funkci $p : \bar{\Omega} \to \mathbb{R}$ splňující následující systém parciálních diferenciálních rovnic:

$$-2\nu \operatorname{div} \mathrm{D}\boldsymbol{u} + \nabla p + \alpha \boldsymbol{u} = \boldsymbol{f} \quad \text{v } \Omega, \tag{22}$$

$$\operatorname{div} \boldsymbol{u} = 0 \quad \text{v } \Omega, \tag{23}$$

$$\boldsymbol{u} = \boldsymbol{u}_D \quad \text{na } \partial\Omega, \tag{24}$$

kde $\mathrm{D}\boldsymbol{u} = \frac{1}{2}(\nabla \boldsymbol{u} + (\nabla \boldsymbol{u}^{\mathsf{T}}))$ je symetrická část gradientu funkce $\boldsymbol{u}$, $\nu > 0$ je viskozita tekutiny, $\alpha > 0$ je parametr související s časovou úlohou, $\boldsymbol{f} : \bar{\Omega} \to \mathbb{R}^d$ je silové pole působící na tekutinu a $\boldsymbol{u}_D : \partial\Omega \to \mathbb{R}^d$ je předepsaná Dirichletova okrajová podmínka. Řešení $\boldsymbol{u}$ představuje rychlostní pole tekutiny v Ω a p je tlakové pole v Ω. Jsou-li funkce $\boldsymbol{f}$ a $\boldsymbol{u}_D$ dostatečně hladké, tj. $\boldsymbol{f} \in (C(\bar{\Omega}))^d$ a $\boldsymbol{u}_D \in (C(\partial\Omega))^d$, pak existuje klasické řešení úlohy (22)-(24). Klasickým řešením přitom nazýváme funkci $\boldsymbol{u} \in (C^2(\Omega))^d$ se spojitým rozšířením na hranici $\partial\Omega$ a funkci $p \in C^1(\Omega)$ takové, že splňují rovnice (22), (23) jako rovnosti, přičemž spojité rozšíření $\boldsymbol{u}$ na $\partial\Omega$ splňuje (24). Rychlostní složka $\boldsymbol{u}$ je určena jednoznačně a tlaková složka p je určena až na aditivní konstantu. Při řešení konkrétních úloh budeme předepisovat hodnoty tlaku p v jednom bodě, čímž docílíme jednoznačnosti řešení také pro tlakovou složku.

Platí následující tvrzení: je-li $\operatorname{div} \boldsymbol{u} = 0$, pak

$$2 \operatorname{div} \mathrm{D}\boldsymbol{u} = \Delta \boldsymbol{u}.$$

Rovnici (22) můžeme proto nahradit rovnicí:

$$-\nu \Delta \boldsymbol{u} + \nabla p + \alpha \boldsymbol{u} = \boldsymbol{f} \quad \text{v } \Omega. \tag{25}$$

Úloha (22)-(24) a úloha (25), (23)-(24) jsou proto plně ekvivalentní. Při výpočtu numerického řešení pomocí smíšené metody konečných prvků budeme aproximovat obě tyto úlohy, jelikož při použití jiných než Dirichletových okrajových podmínek již tato ekvivalence neplatí.

5.2 Slabá formulace úlohy (22)-(24)

Slabou formulaci odvodíme společně pro dvě i tři prostorové dimenze, tj. budeme uvažovat $\boldsymbol{u} = (u_1, u_2)^{\mathsf{T}}$ nebo $\boldsymbol{u} = (u_1, u_2, u_3)^{\mathsf{T}}$. Všimněme si, jak v těchto případech vypadá symetrická

část gradientu $\mathrm{D}\boldsymbol{u}$. Protože gradient vektorové funkce má tvar:

$$\nabla \boldsymbol{u} = \begin{bmatrix} u_{1x} & u_{1y} \\ u_{2x} & u_{2y} \end{bmatrix}, \quad \text{nebo} \qquad \nabla \boldsymbol{u} = \begin{bmatrix} u_{1x} & u_{1y} & u_{1z} \\ u_{2x} & u_{2y} & u_{2z} \\ u_{3x} & u_{3y} & u_{3z} \end{bmatrix},$$

dostáváme pro $d = 2$:

$$\mathrm{D}\boldsymbol{u} = \begin{bmatrix} u_{1x} & (u_{1y} + u_{2x})/2 \\ (u_{1y} + u_{2x})/2 & u_{2y} \end{bmatrix} =: \begin{bmatrix} \mathrm{D}_1\boldsymbol{u} \\ \mathrm{D}_2\boldsymbol{u} \end{bmatrix},$$

a pro $d = 3$:

$$\mathrm{D}\boldsymbol{u} = \begin{bmatrix} u_{1x} & (u_{1y} + u_{2x})/2 & (u_{1z} + u_{3x})/2 \\ (u_{1y} + u_{2x})/2 & u_{2y} & (u_{2z} + u_{3y})/2 \\ (u_{1z} + u_{3x})/2 & (u_{2z} + u_{3y})/2 & u_{3z} \end{bmatrix} =: \begin{bmatrix} \mathrm{D}_1\boldsymbol{u} \\ \mathrm{D}_2\boldsymbol{u} \\ \mathrm{D}_3\boldsymbol{u} \end{bmatrix}.$$

Vektorovou rovnici (22) můžeme rozepsat jako

$$-2\nu \operatorname{div} \mathrm{D}_i\boldsymbol{u} + p_{x_i} + \alpha u_i = f_i \quad \text{v } \Omega, \quad i = 1, \ldots, d, \tag{26}$$

kde $\boldsymbol{f} = (f_1, \ldots, f_d)^{\mathsf{T}}$ a p_{x_i} označuje derivaci podle i-té proměnné, např. $p_{x_2} = \partial p/\partial y$. Každou z rovnic (26) vynásobíme dostatečně hladkou testovací funkci v_i a provedeme integraci přes Ω:

$$-2\nu \int_\Omega \operatorname{div} \mathrm{D}_i\boldsymbol{u} v_i + \int_\Omega p_{x_i} v_i + \alpha \int_\Omega u_i v_i = \int_\Omega f_i v_i, \quad i = 1, \ldots, d. \tag{27}$$

U prvního a druhého integrálu na levé straně (27) použijeme Greenovu formuli:

$$\int_\Omega \operatorname{div} \mathrm{D}_i\boldsymbol{u} v_i = -\int_\Omega \mathrm{D}_i\boldsymbol{u} \cdot \nabla v_i + \int_{\partial\Omega} (\mathrm{D}_i\boldsymbol{u} \cdot \boldsymbol{n}) v_i,$$
$$\int_\Omega p_{x_i} v_i = -\int_\Omega p v_{ix_i} + \int_{\partial\Omega} p v_i n_i,$$

kde $\boldsymbol{n}$ je jednotkový vektor vnější normály k hranici $\partial\Omega$ a n_i je jeho i-tá složka pro $i = 1, \ldots, d$. Dostaneme tak

$$2\nu \int_\Omega \mathrm{D}_i\boldsymbol{u} \cdot \nabla v_i - \int_\Omega p v_{ix_i} + \alpha \int_\Omega u_i v_i = \int_\Omega f_i v_i + \int_{\partial\Omega} (2\nu \mathrm{D}_i\boldsymbol{u} \cdot \boldsymbol{n} - p n_i) v_i \tag{28}$$

pro $i = 1, \ldots, d$. Protože u Dirichletových okrajových podmínek volíme testovací funkce nulové na hranici $\partial\Omega$, vypadne z (28) křivkový integrál. Označíme-li

$$\mathrm{D}\boldsymbol{u} : \nabla \boldsymbol{v} = \sum_{i=1}^{d} \mathrm{D}_i\boldsymbol{u} \cdot \nabla v_i,$$

dostaneme sečtením všechno rovností (28) vztah

$$2\nu \int_\Omega \mathrm{D}\boldsymbol{u} : \nabla \boldsymbol{v} - \int_\Omega p(\nabla \cdot \boldsymbol{v}) + \alpha \int_\Omega \boldsymbol{u} \cdot \boldsymbol{v} = \int_\Omega \boldsymbol{f} \cdot \boldsymbol{v}, \tag{29}$$

který platí pro libovolnou dostatečně hladkou vektorovou funkci $\boldsymbol{v} = (v_1, \ldots, v_d)^\mathsf{T}$. Roznásobením a přezávorkováním lze ukázat, že

$$\mathrm{D}\boldsymbol{u} : \nabla \boldsymbol{v} = \sum_{i=1}^{d} \mathrm{D}_i \boldsymbol{u} \cdot \mathrm{D}_i \boldsymbol{v} =: \mathrm{D}\boldsymbol{u} : \mathrm{D}\boldsymbol{v},$$

takže (29) lze zapsat také ve tvaru:

$$2\nu \int_\Omega \mathrm{D}\boldsymbol{u} : \mathrm{D}\boldsymbol{v} + \alpha \int_\Omega \boldsymbol{u} \cdot \boldsymbol{v} - \int_\Omega p(\nabla \cdot \boldsymbol{v}) = \int_\Omega \boldsymbol{f} \cdot \boldsymbol{v}. \tag{30}$$

Podívejme se nyní na podmínku (23), kterou zapisujeme také takto:

$$\nabla \cdot \boldsymbol{u} = 0 \quad \text{v } \Omega. \tag{31}$$

Jestliže vztah (31) násobíme dostatečně hladkou testovací funkcí q a provedeme integraci přes Ω, dostaneme, že platí:

$$\int_\Omega q(\nabla \cdot \boldsymbol{u}) = 0. \tag{32}$$

Ukázali jsme, že řešení úlohy (22)-(24) splňuje (30) a (32). Budeme-li u řešení i u testovacích funkcí klást co možná nejslabší požadavky na hladkost, dostaneme následující slabou formulaci úlohy (22)-(24):

$$\left.\begin{aligned} &\text{Najdi } \boldsymbol{u} \in (H^1(\Omega))^d,\ \boldsymbol{u} = \boldsymbol{u}_D \text{ na } \partial\Omega \text{ a } p \in L^2(\Omega) \text{ tak, že} \\ &2\nu \int_\Omega \mathrm{D}\boldsymbol{u} : \mathrm{D}\boldsymbol{v} + \alpha \int_\Omega \boldsymbol{u} \cdot \boldsymbol{v} - \int_\Omega p(\nabla \cdot \boldsymbol{v}) = \int_\Omega \boldsymbol{f} \cdot \boldsymbol{v} \quad \forall \boldsymbol{v} \in (H_0^1(\Omega))^d \\ &\qquad\qquad \int_\Omega q(\nabla \cdot \boldsymbol{u}) = 0 \quad \forall q \in L^2(\Omega). \end{aligned}\right\} \ (P)$$

Jestliže je řešení úlohy (P) dostatečně hladké, můžeme pomocí Greenovy věty opačným postupem dokázat, že se jedná rovněž o řešení úlohy (22)-(24).

5.3 Sestavení soustav lineárních rovnic pro úlohu (22)-(24)

V následujících postupech použijeme řadu dílčích výsledků z předchozích kapitol. Zaměříme se především na odvození lokálních matic, eliminaci bublinkových složek a sestavení vektorizovaných kódů.

5.3.1 Dvoudimenzionální případ

Na trojúhelníku T triangulace $\mathcal{T}_h$ oblasti Ω budeme používat bázové funkce

$$\phi_1, \phi_2, \phi_3, \quad \phi_b = 3^3\phi_1\phi_2\phi_3$$

zavedené v odstavcích 3.2 a 3.5. Pro zjednodušení budeme někdy značit $\phi_4 = \phi_b$. Složky vektoru rychlosti $\boldsymbol{u} = (u_1, u_2)^\mathsf{T}$ a tlak p budeme na T aproximovat takto:

$$u_k = \sum_{j=1}^{4} u_{kj}\phi_j, \quad k = 1, 2, \qquad p = \sum_{j=1}^{3} p_j\phi_j.$$

Nejprve odvodíme matici hmotnosti z integrálu:

$$\alpha \int_T \boldsymbol{u} \cdot \boldsymbol{v} = \alpha \int_T u_1 v_1 + \alpha \int_T u_2 v_2,$$

kde $\boldsymbol{v} = (v_1, v_2)^\mathsf{T}$. Pro $\boldsymbol{v} = (\phi_i, 0)^\mathsf{T}$ a $\boldsymbol{v} = (0, \phi_i)^\mathsf{T}$ dostaneme

$$\sum_{j=1}^{4} u_{1j}\alpha \int_T \phi_j\phi_i \quad \text{a} \quad \sum_{j=1}^{4} u_{2j}\alpha \int_T \phi_j\phi_i, \quad i = 1, \ldots, 4,$$

respektive. Lokální matice hmotnosti $\mathbb{M}_T \in \mathbb{R}^{8\times 8}$ má proto tvar:

$$\mathbb{M}_T = \left[\begin{array}{cc|cc} \mathbf{M}_T & \boldsymbol{m}_T & 0 & 0 \\ \boldsymbol{m}_T^\mathsf{T} & \omega_M & 0 & 0 \\ \hline 0 & 0 & \mathbf{M}_T & \boldsymbol{m}_T \\ 0 & 0 & \boldsymbol{m}_T^\mathsf{T} & \omega_M \end{array}\right],$$

kde $\mathbf{M}_T$, $\boldsymbol{m}_T$, ω_M jsou objekty odvozené v odstavcích 3.3 a 3.6:

$$\mathbf{M}_T = \frac{\alpha|T|}{12}\begin{bmatrix} 2 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 2 \end{bmatrix}, \quad \boldsymbol{m}_T = \frac{3\alpha|T|}{20}\begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}, \quad \omega_M = \frac{81\alpha|T|}{280}.$$

Přejděme nyní k lokální matici tuhosti $\mathbb{R}_T \in \mathbb{R}^{8\times 8}$, která je určena integrálem:

$$\begin{aligned} 2\nu \int_T \mathrm{D}\boldsymbol{u} : \mathrm{D}\boldsymbol{v} = 2\nu \int_T \mathrm{D}\boldsymbol{u} : \nabla\boldsymbol{v} = & 2\nu \int_T (u_{1x}v_{1x} + \frac{1}{2}u_{1y}v_{1y}) + \nu \int_T u_{2x}v_{1y} + \\ & + \nu \int_T u_{1y}v_{2x} + 2\nu \int_T (\frac{1}{2}u_{2x}v_{2x} + u_{2y}v_{2y}), \end{aligned}$$

pro $i = 1, \ldots, 4$. Pro $\boldsymbol{v} = (\phi_i, 0)^\mathsf{T}$ dostaneme:

$$\sum_{j=1}^{4} u_{1j} 2\nu \int_T (\phi_{jx}\phi_{ix} + \frac{1}{2}\phi_{jy}\phi_{iy}) + \sum_{j=1}^{4} u_{2j}\nu \int_T \phi_{jx}\phi_{iy},$$

a pro $\boldsymbol{v} = (0, \phi_i)^\mathsf{T}$ dostaneme:

$$\sum_{j=1}^{4} u_{1j}\nu \int_T \phi_{jy}\phi_{ix} + \sum_{j=1}^{4} u_{2j} 2\nu \int_T (\frac{1}{2}\phi_{jx}\phi_{ix} + \phi_{jy}\phi_{iy}).$$

Lokální matice tuhosti $\mathbb{R}_T \in \mathbb{R}^{8\times 8}$ má tento tvar:

$$\mathbb{R}_T = \left[\begin{array}{cc|cc} \mathbf{R}_{T11} & 0 & \mathbf{R}_{T12} & 0 \\ 0 & \omega_{R11} & 0 & \omega_{R12} \\ \hline \mathbf{R}_{T21} & 0 & \mathbf{R}_{T22}^\mathsf{T} & 0 \\ 0 & \omega_{R21} & 0 & \omega_{R22} \end{array}\right],$$

kde

$$(\mathbf{R}_{T11})_{ij} = 2\nu \int_T (\phi_{jx}\phi_{ix} + \frac{1}{2}\phi_{jy}\phi_{iy}), \qquad (\mathbf{R}_{T12})_{ij} = \nu \int_T \phi_{jx}\phi_{iy},$$
$$(\mathbf{R}_{T21})_{ij} = \nu \int_T \phi_{jy}\phi_{ix}, \qquad (\mathbf{R}_{T22})_{ij} = 2\nu \int_T (\frac{1}{2}\phi_{jx}\phi_{ix} + \phi_{jy}\phi_{iy}),$$

pro $i, j = 1, 2, 3$ a

$$\omega_{R11} = 2\nu \int_T (\phi_{bx}\phi_{bx} + \frac{1}{2}\phi_{by}\phi_{by}), \qquad \omega_{R12} = \nu \int_T \phi_{bx}\phi_{by},$$
$$\omega_{R21} = \nu \int_T \phi_{by}\phi_{bx}, \qquad \omega_{R22} = 2\nu \int_T (\frac{1}{2}\phi_{bx}\phi_{bx} + \phi_{by}\phi_{by}).$$

Nulovost mimodiagonálních bloků v submaticích matice $\mathbb{R}_T$ plyne z podobných úvah jakými jsme v odstavci 3.6 dokázali $\mathbf{r}_T = \mathbf{0}$ opírajíce se o Lemma 1. Pro výpočet matic $\mathbf{R}_{Tkl},\ k, l = 1, 2$, použijeme vektory $\boldsymbol{x}_T,\ \boldsymbol{y}_T \in \mathbb{R}^3$ zavedené v odstavci 3.3, pro něž platí:

$$\boldsymbol{x}_T = 2|T|[\phi_{1x}, \phi_{2x}, \phi_{3x}], \quad \boldsymbol{y}_T = 2|T|[\phi_{1y}, \phi_{2y}, \phi_{3y}].$$

Dostaneme:

$$\mathbf{R}_{T11} = \frac{\nu}{2|T|}(\boldsymbol{x}_T\boldsymbol{x}_T^\mathsf{T} + \frac{1}{2}\boldsymbol{y}_T\boldsymbol{y}_T^\mathsf{T}), \quad \mathbf{R}_{T22} = \frac{\nu}{2|T|}(\frac{1}{2}\boldsymbol{x}_T\boldsymbol{x}_T^\mathsf{T} + \boldsymbol{y}_T\boldsymbol{y}_T^\mathsf{T}),$$
$$\mathbf{R}_{T12} = \frac{\nu}{4|T|}\boldsymbol{y}_T\boldsymbol{x}_T^\mathsf{T} = \mathbf{R}_{T21}^\mathsf{T}.$$

Při výpočtu zbývajících členů budeme postupovat podrobněji. Pro zjednodušení zavedeme operátory ∇^x a ∇^y takto: $\nabla^x\phi = (\phi_x, \frac{1}{2}\phi_y)$, $\nabla^y\phi = (\frac{1}{2}\phi_x, \phi_y)$. Tyto operátory mají podobné vlastnosti jako klasický gradient, platí pro ně také analogie Lemmatu 1. Dále platí $\nabla^x\phi \cdot \nabla\psi = \nabla\phi \cdot \nabla^x\psi$

a podobně pro ∇^y. Můžeme psát:

$$\begin{aligned}\omega_{R11} &= 2\nu \int_T \nabla^x \phi_b \cdot \nabla \phi_b = 2\nu 3^6 \int_T \nabla^x (\phi_1\phi_2\phi_3) \cdot \nabla(\phi_1\phi_2\phi_3) = \\ &= 2\nu 3^3 c(\nabla^x \phi_1 \cdot \nabla\phi_1 + \nabla^x \phi_2 \cdot \nabla\phi_2 + \nabla^x \phi_3 \cdot \nabla\phi_3 + \\ &\qquad \nabla^x \phi_1 \cdot \nabla\phi_2 + \nabla^x \phi_1 \cdot \nabla\phi_3 + \\ &\qquad \nabla^x \phi_2 \cdot \nabla\phi_3).\end{aligned}$$

V odstavci 3.6 jsme zjistili, že $c = |T|/90$ a s pomocí Lemmatu 1 můžeme psát:

$$\begin{aligned}\omega_{R11} &= \frac{81\nu|T|}{5}(\nabla^x \phi_1 \cdot \nabla\phi_1 - \nabla^x \phi_2 \cdot \nabla\phi_3) \\ &= \frac{81\nu}{20|T|}(x_{T1}^2 + \frac{1}{2}y_{T1}^2 - x_{T2}x_{T3} - \frac{1}{2}y_{T2}y_{T3}).\end{aligned}$$

Podobně vypočteme také

$$\omega_{R22} = \frac{81\nu}{20|T|}(\frac{1}{2}x_{T1}^2 + y_{T1}^2 - \frac{1}{2}x_{T2}x_{T3} - y_{T2}y_{T3}).$$

Při výpočtu $\omega_{R12} = \omega_{R21}$ použijeme opět konstantu c a Lemma 1:

$$\begin{aligned}\omega_{R12} &= \nu 3^6 \int_T (\phi_1\phi_2\phi_3)_x(\phi_1\phi_2\phi_3)_y = \nu 3^6 \frac{c}{2}(2\phi_{1x}\phi_{1y} + 2\phi_{2x}\phi_{2y} + \\ &\quad + 2\phi_{3x}\phi_{3y} + \phi_{1x}\phi_{2y} + \phi_{2x}\phi_{1y} + \phi_{1x}\phi_{3y} + \phi_{3x}\phi_{1y} + \phi_{2x}\phi_{3y} + \phi_{3x}\phi_{2y}) = \\ &= \frac{81\nu|T|}{20}(\phi_{1x}\phi_{1y} + \phi_{2x}\phi_{2y} + \phi_{3x}\phi_{3y} + (\phi_{1x} + \phi_{2x} + \phi_{3x}) \cdot (\phi_{1y} + \phi_{2y} + \phi_{3y})).\end{aligned}$$

Protože poslední dvě závorky jsou nulové, dostaneme:

$$\omega_{R12} = \omega_{R21} = \frac{81\nu}{80|T|}\boldsymbol{x}_T^\mathsf{T}\boldsymbol{y}_T.$$

Lokální vektor pravé strany vznikne z aproximace integrálu:

$$\int_T \boldsymbol{f} \cdot \boldsymbol{v} = \int_T f_1 v_1 + \int_T f_2 v_2.$$

Postup použitý pro oba integrály na pravé straně bude prakticky stejný jako v odstavcích 3.3 a 3.6, takže

$$\begin{aligned}f_{Ti} &= \frac{1}{3}(f_i(\mathbf{p}_1) + f_i(\mathbf{p}_2) + f_i(\mathbf{p}_3)), \\ \boldsymbol{b}_{Ti} &= \frac{1}{3}|T|f_{Ti}[1,1,1]^\mathsf{T}, \quad b_{bi} = \frac{9}{20}|T|f_{Ti},\end{aligned}$$

kde $\mathbf{p}_1, \mathbf{p}_2$ a $\mathbf{p}_3$ jsou vrcholy trojúhelníka T. Lokální vektor pravé strany má tvar:

$$\mathbb{b}_T = [\boldsymbol{b}_{T1}^\mathsf{T}, b_{b1}, \boldsymbol{b}_{T2}^\mathsf{T}, b_{b2}]^\mathsf{T}.$$

V úloze (P) zbývá vysvětlit aproximaci integrálu s tlakovou složkou v první rovnici a aproximaci druhé rovnice. Nejprve odvodíme lokální divergenční matice pro druhou rovnici. Vyjdeme z výrazu:

$$-\int_T q(u_{1x}+u_{2x}),$$

kam dosadíme aproximace $u_1,\ u_2$ a za q volíme postupně všechny bázové funkce použité při aproximaci talku. Dostaneme:

$$\sum_{j=1}^{4} u_{1j}\left(-\int_T \phi_i\phi_{jx}\right)+\sum_{j=1}^{4} u_{2j}\left(-\int_T \phi_i\phi_{jy}\right).$$

Lokální divergenční matici $\widetilde{\mathbb{B}}_T \in \mathbb{R}^{3\times 8}$ je vhodné rozložit na čtyři části:

$$\widetilde{\mathbb{B}}_T = [\mathbf{B}_{T1}, \mathbf{B}_{1b}, \mathbf{B}_{T2}, \mathbf{B}_{2b}],$$

kde

$$(\mathbf{B}_{T1})_{ij} = -\int_T \phi_i\phi_{jx}, \quad (\mathbf{B}_{T2})_{ij} = -\int_T \phi_i\phi_{jy}, \quad i,j=1,2,3,$$

$$(\mathbf{B}_{1b})_i = -\int_T \phi_i\phi_{bx}, \quad (\mathbf{B}_{2b})_i = -\int_T \phi_i\phi_{by}, \quad i=1,2,3,$$

Prvky matic $\mathbf{B}_{T1}$ a $\mathbf{B}_{T2}$ odvodíme snadno:

$$(\mathbf{B}_{T1})_{ij} = -\phi_{jx}\int_T \phi_i = \frac{-x_{Tj}}{2|T|}\cdot\frac{1}{3}|T| = \frac{-\mathrm{sgn}|\mathbf{X}|}{6}x_{Tj}.$$

Celkově můžeme psát:

$$\mathbf{B}_{T1} = \frac{-\mathrm{sgn}|\mathbf{X}|}{6}[x_T, x_T, x_T]^{\mathsf{T}}, \quad \mathbf{B}_{T2} = \frac{-\mathrm{sgn}|\mathbf{X}|}{6}[y_T, y_T, y_T]^{\mathsf{T}}.$$

Dále ukážeme postup při odvození první složky $\mathbf{B}_{1b}$. Máme:

$$(\mathbf{B}_{1b})_1 = -\int_T \phi_1\phi_{bx} = -27(\phi_{1x}c_1+\phi_{2x}c_2+\phi_{3x}c_3),$$

kde

$$c_1 = \int_T \phi_1\phi_2\phi_3 = \frac{|T|}{60}, \quad c_2 = \int_T \phi_1^2\phi_3 = \frac{|T|}{30} = \int_T \phi_1^2\phi_2 = c_3.$$

Dosazením a použitím Lemmatu 1 odvodíme:

$$(\mathbf{B}_{1b})_1 = -27\frac{|T|}{60}(-\phi_{1x}+2\phi_{1x}+2\phi_{2x}+2\phi_{3x}) = \frac{27\cdot|T|}{60\cdot 2|T|}x_{T1} = \frac{9}{40}\mathrm{sgn}|\mathbf{X}|x_{T1}.$$

Tento výsledek snadno zobecníme, čímž dostaneme:

$$\mathbf{B}_{1b} = \frac{9}{40}\mathrm{sgn}|\mathbf{X}|\boldsymbol{x}_T, \quad \mathbf{B}_{2b} = \frac{9}{40}\mathrm{sgn}|\mathbf{X}|\boldsymbol{y}_T.$$

Nakonec si ještě všimněme integrálu s tlakovou složkou:

$$-\int_T p(\nabla \cdot \boldsymbol{v}) = -\int_T p(v_{1x} + v_{2x}),$$

kde $\boldsymbol{v} = (v_1, v_2)^\mathsf{T}$. Odtud pro $\boldsymbol{v} = (\phi_i, 0)$ a $\boldsymbol{v} = (0, \phi_i)$ dostaneme:

$$\sum_{j=1}^{3} p_j(-\int_T \phi_j \phi_{ix}), \quad \sum_{j=1}^{3} p_j(-\int_T \phi_j \phi_{iy}), \qquad i = 1, \ldots, 4.$$

Je vidět, že lokální gradientní matice je transpozicí lokální divergenční matice. Lokální soustava lineárních rovnic má po permutaci tvar:

$$\begin{bmatrix} \mathbf{A}_T & \mathbf{Z}_T & \mathbf{B}_T^\mathsf{T} \\ \mathbf{Z}_T^\mathsf{T} & \mathbf{W}_T & \mathbf{B}_{bT}^\mathsf{T} \\ \mathbf{B}_T & \mathbf{B}_{bT} & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{u}_T \\ \boldsymbol{u}_{bT} \\ \boldsymbol{p}_T \end{bmatrix} = \begin{bmatrix} \boldsymbol{b}_T \\ \boldsymbol{b}_{bT} \\ 0 \end{bmatrix},$$

kde

$$\mathbf{A}_T = \begin{bmatrix} \mathbf{R}_{T11} + \mathbf{M}_T & \mathbf{R}_{T12} \\ \mathbf{R}_{T21} & \mathbf{R}_{T22} + \mathbf{M}_T \end{bmatrix}, \quad \mathbf{Z}_T = \begin{bmatrix} \boldsymbol{m}_T & 0 \\ 0 & \boldsymbol{m}_T \end{bmatrix},$$

$$\mathbf{B}_T = [\mathbf{B}_{1T}, \mathbf{B}_{2T}], \quad \mathbf{B}_{bT} = [\mathbf{B}_{1b}, \mathbf{B}_{2b}], \quad \mathbf{b}_T = [\mathbf{b}_{T1}^\mathsf{T}, \mathbf{b}_{2T}^\mathsf{T}]^\mathsf{T}, \quad \mathbf{b}_{bT} = [b_{b1}, b_{b2}]^\mathsf{T},$$
$$\boldsymbol{u}_T = [u_{11}u_{12}, u_{13}, u_{21}, u_{22}, u_{23}]^\mathsf{T}, \quad \boldsymbol{u}_{bT} = [u_{1b}, u_{2b}]^\mathsf{T}, \quad \boldsymbol{p} = [p_1, p_2, p_3]^\mathsf{T},$$

a dále $\omega_{ii} = \omega_{Rii} + \omega_M$, $i = 1, 2$, $\omega_{12} = \omega_{21} = \omega_{R12}$ jsou prvky matice $\mathbf{W}$. Pro eliminaci bublinkových kompoment použijeme prostřední blokovou rovnici:

$$\boldsymbol{u}_{bT} = \mathbf{W}_T^{-1}(\boldsymbol{b}_{bT} - \mathbf{Z}_T^\mathsf{T}\boldsymbol{u}_T - \mathbf{B}_{bT}^\mathsf{T}\boldsymbol{p}_T).$$

Pro snadnější vektorizaci výsledného kódu vyjádříme $\mathbf{W}_T^{-1}$ pomocí Cramerova pravidla:

$$\mathbf{W}_T^{-1} = \frac{1}{\det \mathbf{W}_T} \widetilde{\mathbf{W}}_T,$$

kde $\det \mathbf{W}_T = \omega_{11}\omega_{22} - \omega_{12}^2$ a

$$\widetilde{\mathbf{W}}_T = \begin{bmatrix} \omega_{22} & -\omega_{12} \\ -\omega_{12} & \omega_{11} \end{bmatrix}.$$

Po eliminaci v první a třetí blokové rovnici dosaneme tuto lokální soustavu linéárních rovnic:

$$\begin{bmatrix} \mathbb{A}_T & \mathbb{B}_T^\mathsf{T} \\ \mathbb{B}_T & -\mathbb{E} \end{bmatrix} \begin{bmatrix} \boldsymbol{u}_T \\ \boldsymbol{p}_T \end{bmatrix} = \begin{bmatrix} \mathbb{b}_T \\ \mathbb{c}_T \end{bmatrix},$$

kde

$$\mathbb{A}_T = \mathbf{A}_T - \frac{1}{\det \mathbf{W}_T}\mathbf{Z}_T\widetilde{\mathbf{W}}\mathbf{Z}_T^{\mathsf{T}}, \quad \mathbb{B}_T = \mathbf{B}_T - \frac{1}{\det \mathbf{W}_T}\mathbf{B}_{bT}\widetilde{\mathbf{W}}\mathbf{Z}_T^{\mathsf{T}}, \quad \mathbb{E}_T = \frac{1}{\det \mathbf{W}_T}\mathbf{B}_{bT}\widetilde{\mathbf{W}}\mathbf{B}_{bT}^{\mathsf{T}},$$
$$\mathbb{b}_T = \boldsymbol{b}_T - \frac{1}{\det \mathbf{W}_T}\mathbf{Z}_T\widetilde{\mathbf{W}}\boldsymbol{b}_{bT}, \quad \mathbb{c}_T = -\frac{1}{\det \mathbf{W}_T}\mathbf{B}_{bT}\widetilde{\mathbf{W}}\boldsymbol{b}_{bT}.$$

Odvozené vztahy lze použít ke generování globálních matic průchodem přes jednotlivé trojúhelníky anebo vektorizovaným kódem. První přístup ukazuje kód 22, druhý přístup je rozpracován v kódu 23.

```
function [A,B,E,b,c,a]=assemblyP1bP1D(p,t,alpha,nu,f)
np=size(p,1); nt=size(t,1);
A=sparse(2*np,2*np); B=sparse(np,2*np); E=sparse(np,np);
b=zeros(2*np,1); c=zeros(np,1); a=zeros(np,1);
Mel=(1/12)*[2 1 1; 1 2 1; 1 1 2]; % element mass matrix
for ih=1:nt
    itB=t(ih,1:3); it1=2*itB-1; it2=it1+1; it12=[it1,it2];
    x21=p(t(ih,2),1)-p(t(ih,1),1); y12=p(t(ih,1),2)-p(t(ih,2),2);
    x32=p(t(ih,3),1)-p(t(ih,2),1); y23=p(t(ih,2),2)-p(t(ih,3),2);
    x13=p(t(ih,1),1)-p(t(ih,3),1); y31=p(t(ih,3),2)-p(t(ih,1),2);
    tarea=(x21*y31-x13*y12)/2;
    xt=[y23; y31; y12]; yt=[x32; x13; x21];
    mt=(3/20)*alpha*tarea*ones(3,1); Zt=[mt,zeros(3,1);zeros(3,1),mt];
    omega(1,1)=81*(nu/20/tarea*(y23^2+x32^2/2-y31*y12-x13*x21/2)+...
                                                    +alpha*tarea/280);
    omega(2,2)=81*(nu/20/tarea*(y23^2/2+x32^2-y31*y12/2-x13*x21)+...
                                                    +alpha*tarea/280);
    omega(1,2)=81*nu/80/tarea*(yt'*xt);
    omega(2,1)=omega(1,2);
    % element matrices
    omega_det=omega(1,1)*omega(2,2)-omega(1,2)^2;
    omega_bar=[omega(2,2) -omega(1,2); -omega(1,2) omega(1,1)];
    M=alpha*tarea*Mel;
    Ah=[M+nu/2/tarea*(xt*xt'+yt*yt'/2)   nu/4/tarea*(yt*xt')
        nu/4/tarea*(xt*yt')              M+nu/2/tarea*(xt*xt'/2+yt*yt')];
    Ah=Ah-(Zt*omega_bar*Zt')/omega_det;
    Bh =(1/6)*[[xt';xt';xt'],[yt';yt';yt']]; Bbh=(9/40)*[xt,yt];
    Bh=-Bh-(Bbh*omega_bar*Zt')/omega_det;
    Eh=(Bbh*omega_bar*Bbh')/omega_det;
```

```
    % element vectors
    f1t=(f(it1(1))+f(it1(2))+f(it1(3)))/3;
    f2t=(f(it2(1))+f(it2(2))+f(it2(3)))/3;
    fh=tarea/3*[f1t;f1t;f1t;f2t;f2t;f2t]; fbh=9/20*tarea*[f1t;f2t];
    ah=(tarea/3)*[1;1;1]; bh=fh-(Zt*omega_bar*fbh)/omega_det;
    ch=-(Bbh*omega_bar*fbh)/omega_det;
    % assembly global data
    A(it12,it12)=A(it12,it12)+Ah;
    B(itB,it12)=B(itB,it12)+Bh;
    E(itB,itB)=E(itB,itB)+Eh;
    b(it12)=b(it12)+bh;
    c(itB)=c(itB)+ch; a(itB)=a(itB)+ah;
end
```

Kód 22: Standardní sestavení 2D úlohy

```
function [A,B,E,b,c,a]=assemblyP1bP1D_vec(p,t,alpha,nu)
t_even=2*t; t_odd=t_even-1; tt=[t_odd;t_even];
np=size(p,1); np2=2*np;
A=sparse(np2,np2); B=sparse(np,2*np); E=sparse(np,np);
b=zeros(np2,1); c=zeros(np,1); a=zeros(np,1);
x32=p(t(:,3),1)-p(t(:,2),1); y23=p(t(:,2),2)-p(t(:,3),2);
x13=p(t(:,1),1)-p(t(:,3),1); y31=p(t(:,3),2)-p(t(:,1),2);
x21=p(t(:,2),1)-p(t(:,1),1); y12=p(t(:,1),2)-p(t(:,2),2);
tarea=(x21.*y31-x13.*y12)/2; alta=alpha*tarea;
xt=[y23 y31 y12]; yt=[x32 x13 x21];
xt12=xt(:,1).^2; yt12=yt(:,1).^2;
xt2xt3=xt(:,2).*xt(:,3); yt2yt3=yt(:,2).*yt(:,3);
omega11=81/20*nu*(xt12+yt12/2-xt2xt3-yt2yt3/2)./tarea+(81/280)*alta;
omega22=81/20*nu*(xt12/2+yt12-xt2xt3/2-yt2yt3)./tarea+(81/280)*alta;
omega12=81/80*nu*(xt(:,1).*yt(:,1)+xt(:,2).*yt(:,2)+xt(:,3).*yt(:,3))./tarea;
omega_det=omega11.*omega22-omega12.^2;
% Mass matrix
aux=alta/12;
for i=1:3
    for j=1:i
        A=A+sparse(tt(:,i),tt(:,j),[aux;aux],np2,np2);
        A=A+sparse(tt(:,j),tt(:,i),[aux;aux],np2,np2);
    end
end
```

```
% Diffusion matrix
n4ta=(nu/4)./tarea;
for i=1:3
    n4taxi=n4ta.*xt(:,i); n4tayi=n4ta.*yt(:,i);
    for j=i:3 % A11, A22
        n4taxixj=n4taxi.*xt(:,j); n4tayiyj=n4tayi.*yt(:,j);
        aux1=2*n4taxixj+n4tayiyj; aux2=n4taxixj+2*n4tayiyj;
        if i==j
            A=A+sparse(tt(:,i),tt(:,j),[aux1;aux2],np2,np2);
        else
            A=A+sparse(tt(:,i),tt(:,j),[aux1;aux2],np2,np2);
            A=A+sparse(tt(:,j),tt(:,i),[aux1;aux2],np2,np2);
        end
    end
    for j=1:3 % A12, A21
        aux=n4tayi.*xt(:,j);
        A=A+sparse(t_odd(:,i),t_even(:,j),aux,np2,np2);
        A=A+sparse(t_even(:,j),t_odd(:,i),aux,np2,np2);
    end
end
% Divergence matrix
for j=1:3
    aux=[xt(:,j)/6; yt(:,j)/6];
    for i=1:3
        B=B-sparse([t(:,i);t(:,i)],tt(:,j),aux,np,np2);
    end
end
% Bubble components elimination in A
mt=3/20*alta;
aux=mt.*mt./omega_det;
aux12=aux.*omega12;
for i=1:3
    for j=1:3
        A=A-sparse(tt(:,i),tt(:,j),[aux.*omega22;aux.*omega11],np2,np2);
        A=A+sparse(t_odd(:,i),t_even(:,j),aux12,np2,np2);
        A=A+sparse(t_even(:,i),t_odd(:,j),aux12,np2,np2);
    end
end
```

```
% Bubble components elimination in B, E
aux=(9/40)./omega_det; aux11=aux.*omega11; aux22=aux.*omega22; aux12=aux.*
    omega12;
for i=1:3
    aux1=aux11.*yt(:,i)-aux12.*xt(:,i);
    aux1m=aux1.*mt;
    aux2=aux22.*xt(:,i)-aux12.*yt(:,i);
    aux2m=aux2.*mt;
    for j=1:3
        E=E+sparse(t(:,i),t(:,j),(9/40)*(aux1.*yt(:,j)+aux2.*xt(:,j)),np,np);
        B=B-sparse([t(:,i);t(:,i)],tt(:,j),[aux2m;aux1m],np,np2);
    end
end
% Right hand-side vector b
f1=data.f(1:2:np2); f2=data.f(2:2:np2);
f1=tarea.*(f1(t(:,1))+f1(t(:,2))+f1(t(:,3)))/9 ;
f2=tarea.*(f2(t(:,1))+f2(t(:,2))+f2(t(:,3)))/9;
for i=1:3
    b=b+sparse(tt(:,i),1,[f1;f2],np2,1);
end
% Bubble components elimination in b, c and a
aux=(27/20)./omega_det;
f1=aux.*f1;                    f2=aux.*f2;
f1b=omega22.*f1-omega12.*f2;   f2b=-omega12.*f1+omega11.*f2;
f1bm=f1b.*mt;                  f2bm=f2b.*mt;
for i=1:3
    b=b-sparse(tt(:,i),1,[f1bm;f2bm],np2,1);
    c=c-sparse(t(:,i),1,(9/40)*(f1b.*xt(:,i)+f2b.*yt(:,i)),np,1);
    a=a+sparse(t(:,i),1,tarea/3,np,1);
end
```

Kód 23: Vektorizované sestavení 2D úlohy

5.3.2 Třídimenzionální případ

Na čtyřstěnu T dělení $\mathcal{T}_h$ oblasti Ω budeme používat bázové funkce

$$\phi_1, \phi_2, \phi_3, \phi_4, \;\; \phi_b = 4^4 \phi_1 \phi_2 \phi_3 \phi_4$$

zavedené v odstavcích 4.2 a 4.5. Budeme také psát $\phi_5 = \phi_b$. Aproximace složek vektoru rychlosti $\boldsymbol{u} = (u_1, u_2, u_3)^\mathsf{T}$ a tlaku p má tvar:

$$u_k = \sum_{j=1}^{5} u_{kj}\phi_j, \quad k = 1, 2, 3, \qquad p = \sum_{j=1}^{4} p_j \phi_j.$$

V následujícím odvození lokálních matic budeme v maximální možné míře využívat analogii s dvourozměrným případem a výsledky předchozích odstavců. Lokální matice hmotnosti $\mathbb{M}_T \in \mathbb{R}^{15\times 15}$ má tvar:

$$\mathbb{M}_T = \left[\begin{array}{cc|cc|cc} \mathbf{M}_T & \boldsymbol{m}_T & 0 & 0 & 0 & 0 \\ \boldsymbol{m}_T^\mathsf{T} & \omega_M & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & \mathbf{M}_T & \boldsymbol{m}_T & 0 & 0 \\ 0 & 0 & \boldsymbol{m}_T^\mathsf{T} & \omega_M & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & \mathbf{M}_T & \boldsymbol{m}_T \\ 0 & 0 & 0 & 0 & \boldsymbol{m}_T^\mathsf{T} & \omega_M \end{array}\right],$$

kde

$$\mathbf{M}_T = \frac{\alpha|T|}{20}\begin{bmatrix} 2 & 1 & 1 & 1 \\ 1 & 2 & 1 & 1 \\ 1 & 1 & 2 & 1 \\ 1 & 1 & 1 & 2 \end{bmatrix}, \quad \boldsymbol{m}_T = \frac{8\alpha|T|}{105}\begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}, \quad \omega_M = \frac{621\alpha|T|}{3940}.$$

Než napíšeme lokální matici tuhosti $\mathbb{R}_T \in \mathbb{R}^{15\times 15}$, všimněme si jak vypadá příslušný integrál:

$$\begin{aligned} 2\nu \int_T \mathrm{D}\boldsymbol{u} : \mathrm{D}\boldsymbol{v} &= 2\nu \int_T \mathrm{D}\boldsymbol{u} : \nabla \boldsymbol{v} = \\ &= 2\nu \int_T (u_{1x}v_{1x} + \frac{1}{2}u_{1y}v_{1y} + \frac{1}{2}u_{1z}v_{1z}) + \nu \int_T u_{2x}v_{1y} + \nu \int_T u_{3x}v_{1z} + \\ &\quad + \nu \int_T u_{1y}v_{2x} + 2\nu \int_T (\frac{1}{2}u_{2x}v_{2x} + u_{2y}v_{2y} + \frac{1}{2}u_{2z}v_{2z}) + \nu \int_T u_{3y}v_{2z} + \\ &\quad + \nu \int_T u_{1z}v_{3x} + \nu \int_T u_{2z}v_{3y} + 2\nu \int_T (\frac{1}{2}u_{3x}v_{3x} + \frac{1}{2}u_{3y}v_{3y} + u_{3z}v_{3z}). \end{aligned}$$

Takže

$$\mathbb{R}_T = \left[\begin{array}{cc|cc|cc} \mathbf{R}_{T11} & 0 & \mathbf{R}_{T12} & 0 & \mathbf{R}_{T13} & 0 \\ 0 & \omega_{R11} & 0 & \omega_{R12} & 0 & \omega_{R13} \\ \hline \mathbf{R}_{T21} & 0 & \mathbf{R}_{T22} & 0 & \mathbf{R}_{T23} & 0 \\ 0 & \omega_{R21} & 0 & \omega_{R22} & 0 & \omega_{R23} \\ \hline \mathbf{R}_{T31} & 0 & \mathbf{R}_{T32} & 0 & \mathbf{R}_{T33} & 0 \\ 0 & \omega_{R31} & 0 & \omega_{R32} & 0 & \omega_{R33} \end{array}\right],$$

kde

$$\begin{aligned}
\mathbf{R}_{T11} &= \frac{\nu}{18|T|}(\boldsymbol{x}_T\boldsymbol{x}_T^\mathsf{T} + \frac{1}{2}\boldsymbol{y}_T\boldsymbol{y}_T^\mathsf{T} + \frac{1}{2}\boldsymbol{z}_T\boldsymbol{z}_T^\mathsf{T}),\\
\mathbf{R}_{T22} &= \frac{\nu}{18|T|}(\frac{1}{2}\boldsymbol{x}_T\boldsymbol{x}_T^\mathsf{T} + \boldsymbol{y}_T\boldsymbol{y}_T^\mathsf{T} + \frac{1}{2}\boldsymbol{z}_T\boldsymbol{z}_T^\mathsf{T}),\\
\mathbf{R}_{T33} &= \frac{\nu}{18|T|}(\frac{1}{2}\boldsymbol{x}_T\boldsymbol{x}_T^\mathsf{T} + \frac{1}{2}\boldsymbol{y}_T\boldsymbol{y}_T^\mathsf{T} + \boldsymbol{z}_T\boldsymbol{z}_T^\mathsf{T}),\\
\mathbf{R}_{T12} &= \frac{\nu}{36|T|}\boldsymbol{y}_T\boldsymbol{x}_T^\mathsf{T} = \mathbf{R}_{T21}^\mathsf{T},\\
\mathbf{R}_{T13} &= \frac{\nu}{36|T|}\boldsymbol{z}_T\boldsymbol{x}_T^\mathsf{T} = \mathbf{R}_{T31}^\mathsf{T},\\
\mathbf{R}_{T23} &= \frac{\nu}{36|T|}\boldsymbol{z}_T\boldsymbol{y}_T^\mathsf{T} = \mathbf{R}_{T32}^\mathsf{T},
\end{aligned}$$

přičemž jsme použili vektory $\boldsymbol{x}_T, \boldsymbol{y}_T, \boldsymbol{z}_T \in \mathbb{R}^4$ odvozené v odstavci 4.3, pro něž platí:

$$\boldsymbol{x}_T = |\mathbf{X}|[\phi_{1x}, \phi_{2x}, \phi_{3x}, \phi_{4x}]^\mathsf{T}, \quad \boldsymbol{y}_T = |\mathbf{X}|[\phi_{1y}, \phi_{2y}, \phi_{3y}, \phi_{4y}]^\mathsf{T}, \quad \boldsymbol{z}_T = |\mathbf{X}|[\phi_{1z}, \phi_{2z}, \phi_{3z}, \phi_{4z}]^\mathsf{T},$$

kde absolutní hodnota $|\mathbf{X}|$ je $6|T|$.

Odvodit $\omega_{Rkl},\ k,l = 1,2,3$ bude pracnější. Použijeme operátory ∇^x, ∇^y a ∇^z, kde $\nabla^x\phi = [\phi_x, \frac{1}{2}\phi_y, \frac{1}{2}\phi_z]$ a podobně pro ∇^y a ∇^z. Platí $\nabla^x\phi \cdot \nabla\psi = \nabla\phi \cdot \nabla^x\psi$ a podobně pro ∇^y a ∇^z. Ve všech případech také platí analogie Lemmatu 2. Dostáváme:

$$\begin{aligned}
\omega_{R11} &= 2\nu\int_T \nabla^x\phi_b \cdot \nabla\phi_b = 2\nu 4^8\int_T \nabla^x(\phi_1\phi_2\phi_3\phi_4)\cdot\nabla(\phi_1\phi_2\phi_3\phi_4) =\\
&= 2\nu 4^8 c(\nabla^x\phi_1\cdot\nabla\phi_1 + \nabla^x\phi_2\cdot\nabla\phi_2 + \nabla^x\phi_3\cdot\nabla\phi_3 + \nabla^x\phi_4\cdot\nabla\phi_4 +\\
&\qquad \nabla^x\phi_1\cdot\nabla\phi_2 + \nabla^x\phi_1\cdot\nabla\phi_3 + \nabla^x\phi_1\cdot\nabla\phi_4 +\\
&\qquad \nabla^x\phi_2\cdot\nabla\phi_3 + \nabla^x\phi_2\cdot\nabla\phi_4 +\\
&\qquad \nabla^x\phi_3\cdot\nabla\phi_4).
\end{aligned}$$

V odstavci 4.6 jsme zjistili, že $c = |T|/7560$, takže pomocí Lemmatu 2 můžeme psát:

$$\begin{aligned}
\omega_{R11} &= \frac{4^7\nu|T|}{945}(\nabla^x\phi_1\cdot\nabla\phi_1 - \nabla^x\phi_3\cdot\nabla\phi_2 - \nabla^x\phi_4\cdot\nabla\phi_2 - \nabla^x\phi_4\cdot\nabla\phi_3) =\\
&= \frac{4096\nu}{8505|T|}(x_{T1}^2 + \frac{1}{2}y_{T1}^2 + \frac{1}{2}z_{T1}^2 - x_{T2}x_{T3} - \frac{1}{2}y_{T2}y_{T3} - \frac{1}{2}z_{T2}z_{T3} -\\
&\quad - x_{T2}x_{T4} - \frac{1}{2}y_{T2}y_{T4} - \frac{1}{2}z_{T2}z_{T4} - x_{T3}x_{T4} - \frac{1}{2}y_{T3}y_{T4} - \frac{1}{2}z_{T3}z_{T4}).
\end{aligned}$$

Podobně dostaneme s pomocí ∇^y a ∇^z, že platí:

$$\begin{aligned}
\omega_{R22} &= \frac{4096\nu}{8505|T|}(\frac{1}{2}x_{T1}^2 + y_{T1}^2 + \frac{1}{2}z_{T1}^2 - \frac{1}{2}x_{T2}x_{T3} - y_{T2}y_{T3} - \frac{1}{2}z_{T2}z_{T3} -\\
&\quad - \frac{1}{2}x_{T2}x_{T4} - y_{T2}y_{T4} - \frac{1}{2}z_{T2}z_{T4} - \frac{1}{2}x_{T3}x_{T4} - y_{T3}y_{T4} - \frac{1}{2}z_{T3}z_{T4}),
\end{aligned}$$

$$\omega_{R33} = \frac{4096\nu}{8505|T|}(\frac{1}{2}x_{T1}^2 + \frac{1}{2}y_{T1}^2 + z_{T1}^2 - \frac{1}{2}x_{T2}x_{T3} - \frac{1}{2}y_{T2}y_{T3} - z_{T2}z_{T3} -$$
$$- \frac{1}{2}x_{T2}x_{T4} - \frac{1}{2}y_{T2}y_{T4} - z_{T2}z_{T4} - \frac{1}{2}x_{T3}x_{T4} - \frac{1}{2}y_{T3}y_{T4} - z_{T3}z_{T4}).$$

Zbývá se zabývat mimodiagonálními členy. Obdobným postupem jako ve dvoudimenzionálním případě dostaneme:

$$\omega_{R12} = \nu \int_T \phi_{bx}\phi_{by} = \nu 4^8 \int_T (\phi_1\phi_2\phi_3\phi_4)_x(\phi_1\phi_2\phi_3\phi_4)_y =$$
$$= \frac{\nu 4^8 c}{2}(\phi_{1x}\phi_{1y} + \phi_{2x}\phi_{2y} + \phi_{3x}\phi_{3y} + \phi_{4x}\phi_{4y} +$$
$$+(\phi_{1x} + \phi_{2x} + \phi_{3x} + \phi_{4x})(\phi_{1y} + \phi_{2y} + \phi_{3y} + \phi_{4y})).$$

Vzhledem k Lemmatu 2 a s vyčíslením konstanty je:

$$\omega_{R12} = \frac{1024\nu}{8505|T|}\boldsymbol{x}_T^\mathsf{T}\boldsymbol{y}_T.$$

Podobně vypočteme:

$$\omega_{R13} = \frac{1024\nu}{8505|T|}\boldsymbol{x}_T^\mathsf{T}\boldsymbol{z}_T, \quad \omega_{R23} = \frac{1024\nu}{8505|T|}\boldsymbol{y}_T^\mathsf{T}\boldsymbol{z}_T$$

Pro zbývající členy matice tuhosti máme: $\omega_{R21} = \omega_{R12}$, $\omega_{R31} = \omega_{R13}$, $\omega_{R32} = \omega_{R23}$.

Lokální vektor pravé strany vznikne jednoduše rozšířením skalárních výsledku z odstavců 4.3 a 4.6 podobně jako ve dvoudimenzionálním případě. Nyní však máme tři složky $\boldsymbol{f} = (f_1, f_2, f_3)^\mathsf{T}$ takže

$$f_{Ti} = \frac{1}{4}(f_i(\mathbf{p_1}) + f_i(\mathbf{p_2}) + f_i(\mathbf{p_3}) + f_i(\mathbf{p_4})),$$

kde $\mathbf{p_1}, \ldots, \mathbf{p_4}$ jsou vrcholy čtyřstěnu T,

$$\mathbf{b}_{Ti} = \frac{1}{4}|T|f_{Ti}[1,1,1,1]^\mathsf{T}, \quad b_{bi} = \frac{32}{105}|T|f_{Ti},$$

pro $i = 1, 2, 3$. Lokální vektor pravé strany má tvar:

$$\mathbb{b}_T = [\boldsymbol{b}_{T1}^\mathsf{T}, b_{b1}, \boldsymbol{b}_{T2}^\mathsf{T}, b_{b1}, \boldsymbol{b}_{T3}^\mathsf{T}, b_{b1}]^\mathsf{T}.$$

Nakonec stačí odvodit lokální divergentní matici, protože, jak víme ze dvoudimenzionálního případu, lokální gradientní matice je její transpozicí a není ji proto potřeba sestavovat. Začneme příslušným integrálním výrazem

$$-\int_T q(u_{1x} + u_{2y} + u_{3z}),$$

odkud plyne

$$\sum_{j=1}^{5} u_{1j}(-\int_T \phi_i\phi_{jx}) + \sum_{j=1}^{5} u_{2j}(-\int_T \phi_i\phi_{jy}) + \sum_{j=1}^{5} u_{3j}(-\int_T \phi_i\phi_{jz}) \quad i = 1,\dots,4.$$

Lokální divergenční matici $\widetilde{\mathbb{B}}_T \in \mathbb{R}^{4\times 15}$ rozložíme na šest částí:

$$\widetilde{\mathbb{B}}_T = [\mathbf{B}_{T1}, \mathbf{B}_{1b}, \mathbf{B}_{T2}, \mathbf{B}_{2b}, \mathbf{B}_{T3}, \mathbf{B}_{3b}],$$

kde

$$(\mathbf{B}_{T1})_{ij} = -\int_T \phi_i\phi_{jx}, \quad (\mathbf{B}_{T2})_{ij} = -\int_T \phi_i\phi_{jy}, \quad (\mathbf{B}_{T3})_{ij} = -\int_T \phi_i\phi_{jz}, \quad i,j = 1,\dots,4.$$

a dále

$$(\mathbf{B}_{1b})_i = -\int_T \phi_i\phi_{bx}, \quad (\mathbf{B}_{2b})_i = -\int_T \phi_i\phi_{by}, \quad (\mathbf{B}_{3b})_i = -\int_T \phi_i\phi_{bz}, \quad i = 1,\dots,4.$$

Protože

$$(\mathbf{B}_{T1})_{ij} = -\phi_{jx}\int_T \phi_i = -\frac{x_{Tj}}{|\mathbf{X}|}\cdot\frac{1}{4}|T| = -\frac{\mathrm{sgn}|\mathbf{X}|}{24}x_{Tj}$$

a analogický postup lze vykonat i pro $\mathbf{B}_{T2}$ a $\mathbf{B}_{T3}$. Dostáváme:

$$\begin{aligned}
\mathbf{B}_{T1} &= -\frac{\mathrm{sgn}|\mathbf{X}|}{24}[x_T, x_T, x_T, x_T]^\mathsf{T},\\
\mathbf{B}_{T2} &= -\frac{\mathrm{sgn}|\mathbf{X}|}{24}[y_T, y_T, y_T, y_T]^\mathsf{T},\\
\mathbf{B}_{T3} &= -\frac{\mathrm{sgn}|\mathbf{X}|}{24}[z_T, z_T, z_T, z_T]^\mathsf{T}.
\end{aligned}$$

Nyní ukážeme postup odvození první složky $\mathbf{B}_{1b}$ a výsledek pak zobecníme:

$$(\mathbf{B}_{1b})_1 = -\int_T \phi_1\phi_{bx} = -4^4(\phi_{1x}c_1 + \phi_{2x}c_2 + \phi_{3x}c_3 + \phi_{4x}c_4),$$

kde

$$c_1 = \int_T \phi_1\phi_2\phi_3\phi_4 = \frac{3|T|}{2520}, \quad c_2 = \int_T \phi_1^2\phi_3\phi_4 = \frac{3|T|}{1260} = c_3 = c_4.$$

Takže

$$\begin{aligned}
(\mathbf{B}_{1b})_1 &= -4^4\frac{3|T|}{2520}(\phi_{1x} + 2\phi_{2x} + 2\phi_{3x} + 2\phi_{4x}) =\\
&= 4^4\frac{3|T|}{2520}\phi_{1x} = 4^4\frac{3|T|}{2520}\cdot\frac{x_{T1}}{|\mathbf{X}|} = \frac{16}{315}\mathrm{sgn}|\mathbf{X}|x_{T1}.
\end{aligned}$$

Zobecnění pak vypadá takto:

$$\mathbf{B}_{1b} = \frac{16}{315}\mathrm{sgn}|\mathbf{X}|\boldsymbol{x}_T, \quad \mathbf{B}_{2b} = \frac{16}{315}\mathrm{sgn}|\mathbf{X}|\boldsymbol{y}_T, \quad \mathbf{B}_{3b} = \frac{16}{315}\mathrm{sgn}|\mathbf{X}|\boldsymbol{z}_T.$$

Lokální soustava lineárních rovnic je formálně stejná jako ve dvoudimenzionálním případě. Rozdíl spočívá ve velikosti a struktuře jednotlivých bloků. Zmiňme zejména blok $\mathbf{Z}_T$, který vypadá takto:

$$\mathbf{Z}_T = \begin{bmatrix} \boldsymbol{m}_T & 0 & 0 \\ 0 & \boldsymbol{m}_T & 0 \\ 0 & 0 & \boldsymbol{m}_T \end{bmatrix}.$$

Dále si všimněme, že matice $\mathbf{W}_T$ je třetího řádu:

$$\mathbf{W}_T = \begin{bmatrix} \omega_{11} & \omega_{12} & \omega_{13} \\ \omega_{21} & \omega_{22} & \omega_{23} \\ \omega_{31} & \omega_{32} & \omega_{33} \end{bmatrix},$$

takže její inverze odvozená Cramerovým pravidlem je složitější než ve 2D:

$$\mathbf{W}^{-1} = \frac{1}{\det\mathbf{W}}\widetilde{\mathbf{W}},$$

$$\widetilde{\mathbf{W}} = \begin{bmatrix} \omega_{22}\omega_{33} - \omega_{23}^2 & \omega_{23}\omega_{31} - \omega_{21}\omega_{33} & \omega_{21}\omega_{32} - \omega_{22}\omega_{31} \\ \omega_{13}\omega_{32} - \omega_{12}\omega_{33} & \omega_{11}\omega_{33} - \omega_{13}^2 & \omega_{12}\omega_{31} - \omega_{11}\omega_{32} \\ \omega_{12}\omega_{23} - \omega_{13}\omega_{22} & \omega_{13}\omega_{21} - \omega_{11}\omega_{23} & \omega_{11}\omega_{22} - \omega_{12}^2 \end{bmatrix},$$

$$\det\mathbf{W} = \omega_{11}(\widetilde{\mathbf{W}})_{11} + \omega_{12}(\widetilde{\mathbf{W}})_{12} + \omega_{13}(\widetilde{\mathbf{W}})_{13}.$$

Odvozené vztahy použijeme opět pro sestavení globálních matic průchodem přes jednotlivé čtyřstěny anebo pomocí vektorizovaných kódů. V prvním případě se jedná o kód 24 ve druhém případě to je kód 25.

```
function [A,B,E,b,c,a]=assemblyP1bP1D_3(p,t,alpha,nu,f)
np=size(p,1); nt=size(t,1);
A=sparse(3*np,3*np); B=sparse(np,3*np); E=sparse(np,np);
a=zeros(np,1); b=zeros(3*np,1); c=zeros(np,1);
Mel=(1/20)*[2 1 1 1; 1 2 1 1; 1 1 2 1; 1 1 1 2]; % element mass matrix
for ih=1:nt
  itB=t(ih,1:4); it1=3*itB-2; it2=it1+1; it3=it1+2; it123=[it1,it2,it3];
  x21=p(itB(2),1)-p(itB(1),1); y21=p(itB(2),2)-p(itB(1),2);
  x31=p(itB(3),1)-p(itB(1),1); y31=p(itB(3),2)-p(itB(1),2);
  x41=p(itB(4),1)-p(itB(1),1); y41=p(itB(4),2)-p(itB(1),2);
  x32=p(itB(3),1)-p(itB(2),1); y32=p(itB(3),2)-p(itB(2),2);
  x42=p(itB(4),1)-p(itB(2),1); y42=p(itB(4),2)-p(itB(2),2);
  z21=p(itB(2),3)-p(itB(1),3);
  z31=p(itB(3),3)-p(itB(1),3);
```

```
z41=p(itB(4),3)-p(itB(1),3);
z32=p(itB(3),3)-p(itB(2),3);
z42=p(itB(4),3)-p(itB(2),3);
xt=[z32*y42-y32*z42; y31*z41-z31*y41; z21*y41-y21*z41; y21*z31-z21*y31];
yt=[x32*z42-z32*x42; z31*x41-x31*z41; x21*z41-z21*x41; z21*x31-x21*z31];
zt=[y32*x42-x32*y42; x31*y41-y31*x41; y21*x41-x21*y41; x21*y31-y21*x31];
tvolume=(x21*xt(2)+x31*xt(3)+x41*xt(4))/6;
msgn=sign(tvolume); tvolume=abs(tvolume); altv=alpha*tvolume;
ah=0.25*[tvolume;tvolume;tvolume;tvolume];
mt=(alpha*32/105)*ah;
Zt=[mt,zeros(4,2);zeros(4,1),mt,zeros(4,1);zeros(4,2),mt];
xtt=0.5*(xt(1)^2+yt(1)^2+zt(1)^2-xt(2)*(xt(3)+xt(4))-...
         -xt(3)*xt(4)-yt(2)*(yt(3)+yt(4))-yt(3)*yt(4)-...
         -zt(2)*(zt(3)+zt(4))-zt(3)*zt(4));
omegaR=nu*4096/8505/tvolume;
omegaM=(621/3940)*altv;
omega(1,1)=omegaR*(xtt+0.5*(xt(1)^2-xt(2)*(xt(3)+xt(4))-xt(3)*xt(4)))+omegaM;
omega(2,2)=omegaR*(xtt+0.5*(yt(1)^2-yt(2)*(yt(3)+yt(4))-yt(3)*yt(4)))+omegaM;
omega(3,3)=omegaR*(xtt+0.5*(zt(1)^2-zt(2)*(zt(3)+zt(4))-zt(3)*zt(4)))+omegaM;
omega(1,2)=0.25*omegaR*(xt(1)*yt(1)+xt(2)*yt(2)+xt(3)*yt(3));
omega(1,3)=0.25*omegaR*(xt(1)*zt(1)+xt(2)*zt(2)+xt(3)*zt(3));
omega(2,3)=0.25*omegaR*(yt(1)*zt(1)+yt(2)*zt(2)+yt(3)*zt(3));
% element matrices
omega_bar=[omega(2,2)*omega(3,3)-omega(2,3)^2,...
           -omega(2,1)*omega(3,3)+omega(2,3)*omega(3,1),...
            omega(2,1)*omega(3,2)-omega(2,2)*omega(3,1);
           -omega(1,2)*omega(3,3)+omega(1,3)*omega(3,2),...
            omega(1,1)*omega(3,3)-omega(1,3)^2,...
           -omega(1,1)*omega(3,2)+omega(1,2)*omega(3,1);
            omega(1,2)*omega(2,3)-omega(1,3)*omega(2,2),...
           -omega(1,1)*omega(2,3)+omega(1,3)*omega(2,1),...
            omega(1,1)*omega(2,2)-omega(1,2)^2];
omega_det=omega(1,1)*omega_bar(1,1)+omega(1,2)*omega_bar(1,2)+...
                                              +omega(1,3)*omega_bar(1,3);
M=altv*Mel;aux=nu/36/tvolume;
xyzt=0.5*(xt*xt'+yt*yt'+zt*zt');
Ah=[M+2*aux*(xyzt+0.5*xt*xt')  aux*(yt*xt')            aux*(zt*xt')
    aux*(xt*yt')          M+2*aux*(xyzt+0.5*yt*yt')    aux*(zt*yt')
    aux*(xt*zt')               aux*(yt*zt')     M+2*aux*(xyzt+0.5*zt*zt')];
```

```
  Ah=Ah-(Zt*omega_bar*Zt')/omega_det;
  Bh=-msgn/24*[[xt';xt';xt';xt'],[yt';yt';yt';yt'],[zt';zt';zt';zt']];
  Bbh=msgn*16/315*[xt,yt,zt];
  Bh=Bh-(Bbh*omega_bar*Zt')/omega_det;
  Eh=(Bbh*omega_bar*Bbh')/omega_det;
  % element vectors
  f1t=(f(it1(1))+f(it1(2))+f(it1(3))+f(it1(4)))/4;
  f2t=(f(it2(1))+f(it2(2))+f(it2(3))+f(it2(4)))/4;
  f3t=(f(it3(1))+f(it3(2))+f(it3(3))+f(it3(4)))/4;
  fh=tvolume/4*[f1t;f1t;f1t;f1t;f2t;f2t;f2t;f2t;f3t;f3t;f3t;f3t];
  fbh=(32/105)*tvolume*[f1t;f2t;f3t];
  bh=fh-(Zt*omega_bar*fbh)/omega_det;
  ch=-(Bbh*omega_bar*fbh)/omega_det;
  % assembly global data
  A(it123,it123)=A(it123,it123)+Ah;
  B(itB,it123)=B(itB,it123)+Bh;
  E(itB,itB)=E(itB,itB)+Eh;
  a(itB)=a(itB)+ah; b(it123)=b(it123)+bh; c(itB)=c(itB)+ch;
end
```

Kód 24: Standardní sestavení 3D úlohy

```
function [A,B,E,b,c,a]=assemblyP1bP1D_3(p,t,alpha,nu,f)
np=size(p,1); np3=3*np;
tt_3=3*t; tt_2=tt_3-1; tt_1=tt_3-2; tt=[tt_1;tt_2;tt_3];
A=sparse(np3,np3); B=sparse(np,np3); E=sparse(np,np);
b=zeros(np3,1); c=zeros(np,1); a=zeros(np,1);
x21=p(t(:,2),1)-p(t(:,1),1); y21=p(t(:,2),2)-p(t(:,1),2);
x31=p(t(:,3),1)-p(t(:,1),1); y31=p(t(:,3),2)-p(t(:,1),2);
x41=p(t(:,4),1)-p(t(:,1),1); y41=p(t(:,4),2)-p(t(:,1),2);
x32=p(t(:,3),1)-p(t(:,2),1); y32=p(t(:,3),2)-p(t(:,2),2);
x42=p(t(:,4),1)-p(t(:,2),1); y42=p(t(:,4),2)-p(t(:,2),2);
z21=p(t(:,2),3)-p(t(:,1),3);
z31=p(t(:,3),3)-p(t(:,1),3);
z41=p(t(:,4),3)-p(t(:,1),3);
z32=p(t(:,3),3)-p(t(:,2),3);
z42=p(t(:,4),3)-p(t(:,2),3);
xt=[z32.*y42-y32.*z42,y31.*z41-z31.*y41,z21.*y41-y21.*z41,y21.*z31-z21.*y31];
yt=[x32.*z42-z32.*x42,z31.*x41-x31.*z41,x21.*z41-z21.*x41,z21.*x31-x21.*z31];
```

```
zt=[y32.*x42-x32.*y42,x31.*y41-y31.*x41,y21.*x41-x21.*y41,x21.*y31-y21.*x31];
tvolume=(x21.*xt(:,2)+x31.*xt(:,3)+x41.*xt(:,4))/6;
msgn=sign(tvolume); tvolume=abs(tvolume); altv=alpha*tvolume;
xtt=0.5*(xt(:,1).^2+yt(:,1).^2+zt(:,1).^2-xt(:,2).*(xt(:,3)...
    +xt(:,4))-xt(:,3).*xt(:,4)-yt(:,2).*(yt(:,3)+yt(:,4))...
    -yt(:,3).*yt(:,4)-zt(:,2).*(zt(:,3)+zt(:,4))-zt(:,3).*zt(:,4));
omegaR=(nu*4096/8505)./tvolume; omegaM=(621/3940)*altv;
omega11=omegaR.*(xtt+0.5*(xt(:,1).^2-xt(:,2).*(xt(:,3)+xt(:,4))...
                                                -xt(:,3).*xt(:,4)))+omegaM;
omega22=omegaR.*(xtt+0.5*(yt(:,1).^2-yt(:,2).*(yt(:,3)+yt(:,4))...
                                                -yt(:,3).*yt(:,4)))+omegaM;
omega33=omegaR.*(xtt+0.5*(zt(:,1).^2-zt(:,2).*(zt(:,3)+zt(:,4))...
                                                -zt(:,3).*zt(:,4)))+omegaM;
omega12=0.25*omegaR.*(xt(:,1).*yt(:,1)+xt(:,2).*yt(:,2)+xt(:,3).*yt(:,3));
omega13=0.25*omegaR.*(xt(:,1).*zt(:,1)+xt(:,2).*zt(:,2)+xt(:,3).*zt(:,3));
omega23=0.25*omegaR.*(yt(:,1).*zt(:,1)+yt(:,2).*zt(:,2)+yt(:,3).*zt(:,3));
omega_bar11=omega22.*omega33-omega23.^2;
omega_bar12=omega23.*omega13-omega12.*omega33;
omega_bar13=omega12.*omega23-omega22.*omega13;
omega_bar22=omega11.*omega33-omega13.^2;
omega_bar23=omega12.*omega13-omega11.*omega23;
omega_bar33=omega11.*omega22-omega12.^2;
omega_det=omega11.*omega_bar11+omega12.*omega_bar12+omega13.*omega_bar13;
% Mass matrix
aux=altv/20; aux=[aux;aux;aux];
for i=1:4, for j=1:i
  A=A+sparse(tt(:,i),tt(:,j),aux,np3,np3)+sparse(tt(:,j),tt(:,i),aux,np3,np3);
end, end
% Diffusion matrix
n36ta=(nu/36)./tvolume;
for i=1:4
    n36taxi=n36ta.*xt(:,i); n36tayi=n36ta.*yt(:,i); n36tazi=n36ta.*zt(:,i);
    for j=i:4 % A11, A22, A33
        n36taxixj=n36taxi.*xt(:,j);
        n36tayiyj=n36tayi.*yt(:,j);
        n36tazizj=n36tazi.*zt(:,j);
        aux1=2*n36taxixj+n36tayiyj+n36tazizj;
        aux2=n36taxixj+2*n36tayiyj+n36tazizj;
        aux3=n36taxixj+n36tayiyj+2*n36tazizj;
```

```
        if i==j
            A=A+sparse(tt(:,i),tt(:,j),[aux1;aux2;aux3],np3,np3);
        else
            A=A+sparse(tt(:,i),tt(:,j),[aux1;aux2;aux3],np3,np3);
            A=A+sparse(tt(:,j),tt(:,i),[aux1;aux2;aux3],np3,np3);
        end
    end
    for j=1:4 % A12, A21, A23
        aux=n36tayi.*xt(:,j);
        A=A+sparse(tt_1(:,i),tt_2(:,j),aux,np3,np3);
        A=A+sparse(tt_2(:,j),tt_1(:,i),aux,np3,np3);
        aux=n36tazi.*xt(:,j);
        A=A+sparse(tt_1(:,i),tt_3(:,j),aux,np3,np3);
        A=A+sparse(tt_3(:,j),tt_1(:,i),aux,np3,np3);
        aux=n36tazi.*yt(:,j);
        A=A+sparse(tt_2(:,i),tt_3(:,j),aux,np3,np3);
        A=A+sparse(tt_3(:,j),tt_2(:,i),aux,np3,np3);
    end
end
% Ddivergence matrix
msgn24=msgn./24;
for j=1:4
    aux=[msgn24.*xt(:,j);msgn24.*yt(:,j);msgn24.*zt(:,j)];
    for i=1:4
        B=B-sparse([t(:,i);t(:,i);t(:,i)],tt(:,j),aux,np,np3);
    end
end
% Bubble components elimination in A
mt=(8/105)*altv; aux=mt.*mt./omega_det;
aux12=aux.*omega_bar12; aux13=aux.*omega_bar13; aux23=aux.*omega_bar23;
aux=[aux.*omega_bar11;aux.*omega_bar22;aux.*omega_bar33];
for i=1:4, for j=1:4
    A=A-sparse(tt(:,i),tt(:,j),aux,np3,np3);
    A=A-sparse(tt_1(:,i),tt_2(:,j),aux12,np3,np3);
    A=A-sparse(tt_2(:,i),tt_1(:,j),aux12,np3,np3);
    A=A-sparse(tt_1(:,i),tt_3(:,j),aux13,np3,np3);
    A=A-sparse(tt_3(:,i),tt_1(:,j),aux13,np3,np3);
    A=A-sparse(tt_2(:,i),tt_3(:,j),aux23,np3,np3);
    A=A-sparse(tt_3(:,i),tt_2(:,j),aux23,np3,np3);
```

```
end, end
% Bubble components elimination in B, E
aux=(16/315)./omega_det; msgnmt=msgn.*mt;
aux11=aux.*omega_bar11; aux22=aux.*omega_bar22; aux33=aux.*omega_bar33;
aux12=aux.*omega_bar12; aux13=aux.*omega_bar13; aux23=aux.*omega_bar23;
for i=1:4
    aux1=aux11.*xt(:,i)+aux12.*yt(:,i)+aux13.*zt(:,i); aux1m=aux1.*msgnmt;
    aux2=aux12.*xt(:,i)+aux22.*yt(:,i)+aux23.*zt(:,i); aux2m=aux2.*msgnmt;
    aux3=aux13.*xt(:,i)+aux23.*yt(:,i)+aux33.*zt(:,i); aux3m=aux3.*msgnmt;
    for j=1:4
        E=E+sparse(t(:,i),t(:,j),(16/315)*(aux1.*xt(:,j)+aux2.*yt(:,j)...
                                                    +aux3.*zt(:,j)),np,np);
        B=B-sparse([t(:,i);t(:,i);t(:,i)],tt(:,j),[aux1m;aux2m;aux3m],np,np3);
    end
end
% Right hand-side vector b
f1=data.f(1:3:np3-2); f2=data.f(2:3:np3-1); f3=data.f(3:3:np3);
f1=tvolume.*(f1(t(:,1))+f1(t(:,2))+f1(t(:,3))+f1(t(:,4)))/16;
f2=tvolume.*(f2(t(:,1))+f2(t(:,2))+f2(t(:,3))+f2(t(:,4)))/16;
f3=tvolume.*(f3(t(:,1))+f3(t(:,2))+f3(t(:,3))+f3(t(:,4)))/16;
for i=1:4
    b=b+sparse(tt(:,i),1,[f1;f2;f3],np3,1);
end
% Bubble components elimination in b, c and a
aux=(128/105)./omega_det;
f1=aux.*f1; f2=aux.*f2; f3=aux.*f3;
f1b=omega_bar11.*f1+omega_bar12.*f2+omega_bar13.*f3;
f2b=omega_bar12.*f1+omega_bar22.*f2+omega_bar23.*f3;
f3b=omega_bar13.*f1+omega_bar23.*f2+omega_bar33.*f3;
f1bm=f1b.*mt; f2bm=f2b.*mt; f3bm=f3b.*mt; aux=(16/315)*msgn;
for i=1:4
    b=b-sparse(tt(:,i),1,[f1bm;f2bm;f3bm],np3,1);
    c=c-sparse(t(:,i),1,aux.*(f1b.*xt(:,i)+f2b.*yt(:,i)+f3b.*zt(:,i)),np,1);
    a=a+sparse(t(:,i),1,tvolume/4,np,1);
end
```

Kód 25: Vektorizované sestavení 3D úlohy

5.4 Slabá formulace úlohy (25),(23)-(24)

Slabou formulaci úlohy (25),(23)-(24) odvodíme pomocí Greenovy formule analogickým postupem jako v odstavci 5.2. Odvozování bude jednodušší, protože namísto operátoru 2divD budeme pracovat s vektorovým laplaciánem Δ. Ve skalárních případech pro úlohy (4) a (15) jsme podstatnou část odvození ukázali v dodatku A.2.

Slabá formulace úlohy (25),(23)-(24) má tento tvar:

$$\left.\begin{aligned} &\text{Najdi } \boldsymbol{u} \in (H^1(\Omega))^d,\ \boldsymbol{u} = \boldsymbol{u}_D \text{ na } \partial\Omega \text{ a } p \in L^2(\Omega) \text{ tak, že} \\ &\nu \int_\Omega \nabla \boldsymbol{u} : \nabla \boldsymbol{v} + \alpha \int_\Omega \boldsymbol{u} \cdot \boldsymbol{v} - \int_\Omega p(\nabla \cdot \boldsymbol{v}) = \int_\Omega \boldsymbol{f} \cdot \boldsymbol{v} \quad \forall \boldsymbol{v} \in (H_0^1(\Omega))^d \\ &\int_\Omega q(\nabla \cdot \boldsymbol{u}) = 0 \quad \forall q \in L^2(\Omega). \end{aligned}\right\} (P)$$

5.5 Sestavení soustav lineárních rovnic pro úlohu (25),(23)-(24)

Jednotlivé integrály z poslední úlohy (P) budeme aproximovat stejnými postupy jako v odstavcích 5.3.1 a 5.3.2, vyjma prvního integrálu v první rovnici. Z jeho vyjádření např pro $d = 3$ na čtyřstěnu T,

$$\nu \int_T \nabla \boldsymbol{u} : \nabla \boldsymbol{v} = \nu \int_T \nabla \boldsymbol{u}_1 \cdot \nabla \boldsymbol{v}_1 + \nu \int_T \nabla \boldsymbol{u}_2 \cdot \nabla \boldsymbol{v}_2 + \nu \int_T \nabla \boldsymbol{u}_3 \cdot \nabla \boldsymbol{v}_3,$$

je patrné, že lokální matice tuhosti $\mathbb{R}_T \in \mathbb{R}^{15\times 15}$ bude blokově diagonální, všechny tři bloky budou stejné a jejich odvození je provedeno pro skalární úlohu v odstavci 4.6. Poznamenejme ještě, že výrazně jednodušší bude eliminace bublinkových komponent, protože $\mathbf{W}_T = \omega_R \mathbf{I} \in \mathbb{R}^{15\times 15}$ a $\mathbf{W}_T^{-1} = \omega_R^{-1}\mathbf{I}$, kde ω_R je konstanta z odstavce 4.6.

Kódy pro generování globálních matic tuhosti jsou uvedeny v dodatku E. Kódy pro 2D případ jsou vytvořeny podle práce [7].

6 Závěr

Cílem práce bylo seznámit se s vytvářením sítí a matic tuhosti při řešení diferenciálních rovnic metodou konečných prvků. Sítě byly vytvářeny v jedné až třech prostorových dimenzích. V jedné dimenzi byla síť vytvořena pouhým rozdělením intervalu na menší subintervaly. Ve dvou dimenzích byly vytvořeny sítě o různé hustotě pro několik útvarů s pomocí volného balíku MESH2D. Ve třech dimenzích byl využit balík ISO2MESH na vytvoření sítí krychle, kvádru, L-tvaru a válce.

Na takto vytvořených sítích byla následně testována výpočetní náročnost různých způsobů sestavení matic tuhosti. Především se jednalo o standardní a vektorizované sestavení. U vícedimenzionálních úloh byla navíc testována náročnost sestavení při použití aproximace obsahující bublinkové funkce.

Zjištěný rozdíl mezi výpočetními nároky byl značný. U 1D případu byla výpočetní náročnost vektorizovaného sestavení pro méně než deset prvků asi poloviční ve srovnání se standardním sestavením. Při 1500 prvcích už vektorizované sestavení zabralo méně než procento výpočetní doby standardního sestavení. Pro 98304 elementů to již bylo méně než 0.02%.

Obdobné výsledky vyšly i u dvoudimenzionálního případu. Zde bylo testováno sedm různých útvarů pro různé hustoty sítě. Ve všech případech byl výsledný poměr časů sestavení podobný a pro počet elementů větší než 50000 potřebovalo vektorizované sestavení méně než procento výpočetního času. Dále vyšlo najevo, že tvar oblastí podobných velikostí nemá při konstantním počtu prvků vliv na výpočetní náročnost.

Náročnost sestavení s bublinkovými funkcemi byla testována na vektorizovaných kódech pro řádově 10^4 až 10^6 prvků. Zde vyšlo najevo, že výpočetní náročnost vektorizovaného sestavení s bublinkovými funkcemi je o 3-5 % náročnější než lineárními bázovými funkcemi.

U třech dimenzích pokračoval obdobný trend, jako v předchozích případech. Výpočetní náročnost standardního sestavení byla mnohem větší než při vektorizovaném sestavení. Při testování náročnosti sestavení s bublinkovými funkcemi byl při počtu elementů 10^5 až 10^7 zjištěn nárůst výpočetního času o 4-8 %.

Během vytváření sítí byl balík MESH2D při větším počtu prvků pomalý ve srovnání s balíkem ISO2MESH, kde byl potřebný čas prakticky zanedbatelný a proto jediným omezením byla velikost paměti RAM.

7 Seznam použité literatury

[1] MESH2D, volný toolbox v Matlabu,
https://github.com/dengwirda/mesh2d.

[2] ISO2MESH, volný toolbox pro Matlab,
http://iso2mesh.sourceforge.net/cgi-bin/index.cgi.

[3] R. Blaheta: Matematické modelování a metoda konečných prvků, VŠB-TUO, Ostrava, 2012
http://mi21.vsb.cz/sites/mi21.vsb.cz/files/unit/numericke_metody_2.pdf.

[4] J. Koko: Vectorized Matlab Codes for the Stokes Problem with P1-Bubble/P1 Finite Element,
https://www.isima.fr/~jkoko/Codes/StokesP1BubbleP1.pdf.

A Odvození slabé formulace

A.1 1D úloha

Rovnici (1) vynásobíme dostatečně hladkou testovací funkcí v a integrujeme přes interval (a, b):

$$-\int_a^b u''v = \int_a^b fv.$$

Levou stranu této rovnice integrujeme s užitím per partes

$$\int_a^b u'v' - u'(b)v(b) + u'(a)v(a) = \int_a^b fv.$$

Uvažujeme-li funkce v takové, že $v(a) = v(b) = 0$, dostaneme, že řešení úlohy (1) splňuje

$$\int_a^b (u'v') = \int_a^b fv \quad \forall v \in H_0^1((a,b)),$$
$$u(a) = u_a,\ u(b) = u_b.$$

A.2 2D a 3D úloha

Rovnici (4) resp. (15) vynásobíme dostatečně hladkou testovací funkcí v a integrujeme přes oblast $\Omega \subset \mathbb{R}^d,\ d = 2, 3$:

$$-\int_\Omega \Delta uv + \alpha \int_\Omega uv = \int_\Omega fv.$$

Pro první integrál na levé straně této rovnice použijeme Greenovu formuli:

$$\int_\Omega \Delta uv = -\int_\Omega \nabla u \cdot \nabla v + \int_{\partial\Omega} \frac{\mathrm{d}u}{\mathrm{d}\boldsymbol{n}} v,$$

kde $\boldsymbol{n} \in \mathbb{R}^d,\ d = 2, 3$, je jednotkový vektor vnější normály k $\partial\Omega$. Uvažujeme-li funkce v takové, že $v = 0$ na $\partial\Omega$, křivkový integrál vypadne a zjišťujeme, že řešení úlohy (4) splňuje tyto vztahy:

$$\int_\Omega \nabla u \cdot \nabla v + \alpha \int_\Omega uv = \int_\Omega fv \quad \forall v \in H_0^1(\Omega),$$
$$u = g \quad \text{na } \partial\Omega.$$

Uvedený postup je formálně stejný ve 2D i 3D případě, tj. pro $d = 2$ i $d = 3$.

B Integrály

B.1 2D úloha

Ukažme například výpočet integrálu z diagonály:

$$\int_{\widehat{T}} \widehat{\phi}_2(\boldsymbol{\xi})\widehat{\phi}_2(\boldsymbol{\xi})\,\mathrm{d}\boldsymbol{\xi} = \int_0^1 \int_0^{1-\xi_2} \xi_2^2\,\mathrm{d}\xi_1\,\mathrm{d}\xi_2 = \int_0^1 \xi_2^2[\xi_1]_0^{1-\xi_2}\,\mathrm{d}\xi_2$$
$$= \int_0^1 \xi_2^2 - \xi_2^3\,\mathrm{d}\xi_2 = \left[\frac{\xi_2^3}{3} - \frac{\xi_2^4}{4}\right]_0^1 = \frac{1}{3} - \frac{1}{4} = \frac{1}{12}.$$

Lze snadno dokázat, že

$$\int_{\widehat{T}} \widehat{\phi}_2(\boldsymbol{\xi})\widehat{\phi}_2(\boldsymbol{\xi})\,\mathrm{d}\boldsymbol{\xi} = \int_{\widehat{T}} \widehat{\phi}_1(\boldsymbol{\xi})\widehat{\phi}_1(\boldsymbol{\xi})\,\mathrm{d}\boldsymbol{\xi} = \int_{\widehat{T}} \widehat{\phi}_3(\boldsymbol{\xi})\widehat{\phi}_3(\boldsymbol{\xi})\,\mathrm{d}\boldsymbol{\xi} = \frac{1}{12}.$$

Dále ukažme výpočet integrálu:

$$\int_{\widehat{T}} \widehat{\phi}_2(\boldsymbol{\xi})\widehat{\phi}_1(\boldsymbol{\xi})\,\mathrm{d}\boldsymbol{\xi} = \int_0^1 \int_0^{1-\xi_1} \xi_2\xi_1\,\mathrm{d}\xi_2\,\mathrm{d}\xi_1 = \int_0^1 \xi_1\frac{1-2\xi_1+\xi_1^2}{2}\,\mathrm{d}\xi_1 =$$
$$= \int_0^1 \frac{\xi_1}{2} - \xi_1^2 + \frac{\xi_1^3}{2}\,\mathrm{d}\xi_1 = \left[\frac{\xi_1^2}{4} - \frac{\xi_1^3}{3} + \frac{\xi_1^4}{8}\right]_0^1 = \frac{1}{4} - \frac{1}{3} + \frac{1}{8} = \frac{1}{24}.$$

Vypočtením zbývajících integrálů zjistíme, že platí:

$$\int_{\widehat{T}} \widehat{\phi}_j(\boldsymbol{\xi})\widehat{\phi}_i(\boldsymbol{\xi}) = \frac{1}{24}, \quad \text{pro } i \neq j.$$

Dosazením výsledků do vztahu (8) získáme vztah (9).
Jako další uveďme výpočet integrálů:

$$\int_T \phi_2^2\phi_3^2\,\mathrm{d}\boldsymbol{x} = 2|T|\int_{\widehat{T}} \widehat{\phi}_1^2\widehat{\phi}_2^2\,\mathrm{d}\boldsymbol{\xi} = 2|T|\int_0^1\int_0^{1-\xi_1} \xi_1^2\xi_2^2\,\mathrm{d}\xi_2\,\mathrm{d}\xi_1 =$$
$$= 2|T|\int_0^1 \xi_1^2\left(\frac{1}{3} - \xi_1 + \xi_1^2 - \frac{\xi_1^3}{3}\right)\mathrm{d}\xi_1 = 2|T|\left[\frac{\xi_1^3}{9} - \frac{\xi_1^4}{4} + \frac{\xi_1^5}{5} - \frac{\xi_1^6}{18}\right]_0^1 =$$
$$= 2|T|\left(\frac{1}{9} - \frac{1}{4} + \frac{1}{5} - \frac{1}{18}\right) = \frac{|T|}{90},$$

$$\int_T \phi_1\phi_2^2\phi_3\,\mathrm{d}\boldsymbol{x} = 2|T|\int_{\widehat{T}} \widehat{\phi}_0\widehat{\phi}_1^2\widehat{\phi}_2\,\mathrm{d}\boldsymbol{\xi} = 2|T|\int_0^1 \xi_1^2\int_0^{1-\xi_1}(1-\xi_1-\xi_2)\xi_2\,\mathrm{d}\xi_2\,\mathrm{d}\xi_1 =$$
$$= 2|T|\int_0^1 \xi_1^2\left[\frac{\xi_2^2}{2} - \xi_1\frac{\xi_2^2}{2} - \frac{\xi_2^3}{3}\right]_0^{1-\xi_1}\mathrm{d}\xi_1 = 2|T|\int_0^1 \frac{\xi_1^2}{6} - \frac{\xi_1^3}{2} + \frac{\xi_1^4}{2} - \frac{\xi_1^5}{6}\,\mathrm{d}\xi_1 =$$
$$= 2|T|\left(\frac{1}{18} - \frac{1}{8} + \frac{1}{10} - \frac{1}{36}\right) = \frac{|T|}{180}.$$

Výsledné integrály jsou v následující tabulce:

$\int_{\widehat{T}} \widehat{\phi}_i \widehat{\phi}_i = \frac{1}{12}$	$\int_{\widehat{T}} \widehat{\phi}_i \widehat{\phi}_j = \frac{1}{24} \quad i \neq j$
$\int_{\widehat{T}} \widehat{\phi}_i^2 \widehat{\phi}_j^2 = \frac{1}{180} \quad i \neq j$	$\int_{\widehat{T}} \widehat{\phi}_i \widehat{\phi}_j \widehat{\phi}_k = \frac{1}{120} \quad i \neq j \neq k$
$\int_{\widehat{T}} \widehat{\phi}_i^2 \widehat{\phi}_j \widehat{\phi}_k = \frac{1}{360} \quad i \neq j \neq k$	$\int_{\widehat{T}} \widehat{\phi}_i^2 \widehat{\phi}_j^2 \widehat{\phi}_k^2 = \frac{1}{5040} \quad i \neq j \neq k$

Dále uveďme některé další prvky, jako příkazy pro výpočet v SYMBOLICu:

```
syms x y
syms f0(x,y) f1(x,y) f2(x,y)
f0(x,y) = 1-x-y;
f1(x,y) = x;
f2(x,y) = y;
% Výpočet bloku (m_T)1
mT_1=int(int(f0^2*f1*f2,y,0,1-x),x,0,1)
% Výpočet Omegy_M
omega_M=int(int(f0^2*f1^2*f2^2,y,0,1-x),x,0,1)
% Výpočet složky b_B
b_b = int(int(3^3*(f0*f1*f2),y,0,1-x),x,0,1)
```

B.2 3D úloha

Provedeme výpočet integrálu z diagonály:

$$\begin{aligned}\int_{\widehat{T}} \widehat{\phi}_1(\boldsymbol{\xi})\widehat{\phi}_1(\boldsymbol{\xi})\,\mathrm{d}\boldsymbol{\xi} &= \int_0^1 \int_0^{1-\xi_1} \int_0^{1-\xi_1-\xi_2} \xi_1^2 \,\mathrm{d}\xi_3\,\mathrm{d}\xi_2\,\mathrm{d}\xi_1 = \int_0^1 \xi_1^2 \int_0^{1-\xi_1} 1-\xi_1-\xi_2\,\mathrm{d}\xi_2\,\mathrm{d}\xi_1 = \\ &= \int_0^1 \xi_1^2 \left(1-\xi_1-\xi_1(1-\xi_1)-\frac{(1-\xi_1)^2}{2}\right)\mathrm{d}\xi_1 = \int_0^1 \frac{\xi_1^2}{2} - \xi_1^3 + \frac{\xi_1^4}{2}\,\mathrm{d}\xi_1 = \\ &= \left[\frac{\xi_1^3}{6} - \frac{\xi_1^4}{4} + \frac{\xi_1^5}{10}\right]_0^1 = \frac{1}{6} - \frac{1}{4} + \frac{1}{10} = \frac{1}{60}.\end{aligned}$$

Obdobně jako ve 2D lze dokázat, že

$$\int_{\widehat{T}} \widehat{\phi}_2(\boldsymbol{\xi})\widehat{\phi}_2(\boldsymbol{\xi}) = \frac{1}{60}, \quad \text{pro } i = j.$$

Jako další vypočteme integrál:

$$\int_{\widehat{T}} \widehat{\phi}_1(\boldsymbol{\xi})\widehat{\phi}_2(\boldsymbol{\xi})\,\mathrm{d}\boldsymbol{\xi} = \int_0^1 \int_0^{1-\xi_1} \int_0^{1-\xi_1-\xi_2} \xi_1\xi_2\,\mathrm{d}\xi_3\,\mathrm{d}\xi_2\,\mathrm{d}\xi_1 = \int_0^1 \xi_1 \int_0^{1-\xi_1} \xi_2(1-\xi_1-\xi_2)\,\mathrm{d}\xi_2\,\mathrm{d}\xi_1 =$$
$$= \int_0^1 \xi_1 \left[\frac{\xi_2^2}{2} - \frac{\xi_1\xi_2^2}{2} - \frac{\xi_2^3}{3}\right]_0^{1-\xi_1} \mathrm{d}\xi_1 = \int_0^1 \xi_1 \left(\frac{1}{6} - \frac{\xi_1}{2} + \frac{\xi_1^2}{2} - \frac{\xi_1^3}{6}\right) \mathrm{d}\xi_1 =$$
$$= \left[\frac{\xi_1^2}{12} - \frac{\xi_1^3}{6} + \frac{\xi_1^4}{8} - \frac{\xi_1^5}{30}\right]_0^1 = \frac{1}{12} - \frac{1}{6} + \frac{1}{8} - \frac{1}{30} = \frac{6}{720} = \frac{1}{120}.$$

Zde také platí vztah:

$$\int_{\widehat{T}} \widehat{\phi}_j(\boldsymbol{\xi})\widehat{\phi}_i(\boldsymbol{\xi}) = \frac{1}{120}, \quad \text{pro } i \neq j,$$

takže po dosazení výsledků do jednotlivých složek matice $\mathbf{M}_T$ získáme vztah 18.
Výsledné integrály jsou v následující tabulce:

$\int_{\widehat{T}} \widehat{\phi}_i = \frac{1}{24}$	$\int_{\widehat{T}} \widehat{\phi}_i\widehat{\phi}_i = \frac{1}{60}$
$\int_{\widehat{T}} \widehat{\phi}_i\widehat{\phi}_j = \frac{1}{120} \quad i \neq j$	$\int_{\widehat{T}} \widehat{\phi}_i\widehat{\phi}_j\widehat{\phi}_k = \frac{1}{720} \quad i \neq j \neq k$
$\int_{\widehat{T}} \widehat{\phi}_i\widehat{\phi}_j\widehat{\phi}_k\widehat{\phi}_l = \frac{1}{5040} \quad i \neq j \neq k \neq l$	$\int_{\widehat{T}} \widehat{\phi}_i^2\widehat{\phi}_j\widehat{\phi}_k\widehat{\phi}_l = \frac{1}{20160} \quad i \neq j \neq k \neq l$
$\int_{\widehat{T}} \widehat{\phi}_i^2\widehat{\phi}_j^2\widehat{\phi}_k\widehat{\phi}_l = \frac{1}{90720} \quad i \neq j \neq k \neq l$	$\int_{\widehat{T}} \widehat{\phi}_i^2\widehat{\phi}_j^2\widehat{\phi}_k^2\widehat{\phi}_l^2 = \frac{1}{2494800} \quad i \neq j \neq k \neq l$

Dále uveďme některé další prvky, jako příkazy pro výpočet v SYMBOLICu:

```
syms x y
z~syms f0(x,y,z) f1(x,y,z) f2(x,y,z) f3(x,y,z)
f0(x,y,z) = 1-x-y-z;
f1(x,y,z) = x;
f2(x,y,z) = y;
f3(x,y,z) = z;
% Výpočet bloku (m_T)1
m3D = int(int(int(f0^2*f1*f2*f3,z,0,1-x-y),y,0,1-x),x,0,1)
% Výpočet Omegy_M
omega_M=int(int(int(f0^2*f1^2*f2^2*f3^2,z,0,1-x-y),y,0,1-x),x,0,1)
% Výpočet konstanty c_1
c_1=int(int(int(f0*f1*f2,z,0,1-x-y),y,0,1-x),x,0,1)
% Výpočet konstanty c_2
c_2=int(int(int(f0^2*f1^2*f2*f3,z,0,1-x-y),y,0,1-x),x,0,1)
% Výpočet složky b_B
b_b = int(int(int(f0*f1*f2*f3,z,0,1-x-y),y,0,1-x),x,0,1)
```

C MESH2D

C.1 Základní informace

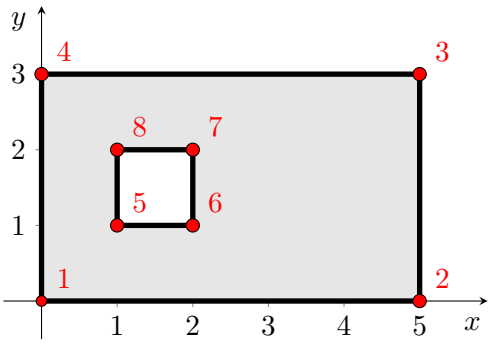
Balík MESH2D od Darren Engwirda, generuje nestrukturované sítě pro obecné dvourozměrné geometrie a je dostupný na

https://github.com/dengwirda/mesh2d

C.2 Zadání geometrie oblasti a generování sítě

Oblast s polygonální hranicí zadáme pomocí souřadnic jednotlivých uzlů ležících na této hranici. Proto např. oblast na obrázku 15 popíšeme čtyřmi uzly na vnějším obdélníku a čtyřmi uzly na vnitřním čtverci. Do vstupní proměnné `node` uvedeme x, y souřadnice těchto uzlů. Hrany tvořící hranici oblasti uvedeme do proměnné `edge`, kde každou hranu popíšeme indexy krajních bodů.

V posledním řádku skriptu je uvedeno volání funkce `refine2`, která vygeneruje síť. Její hustotu upřesníme pomocí parametru `h`.



Obrázek 15: Zadání geometrie oblasti

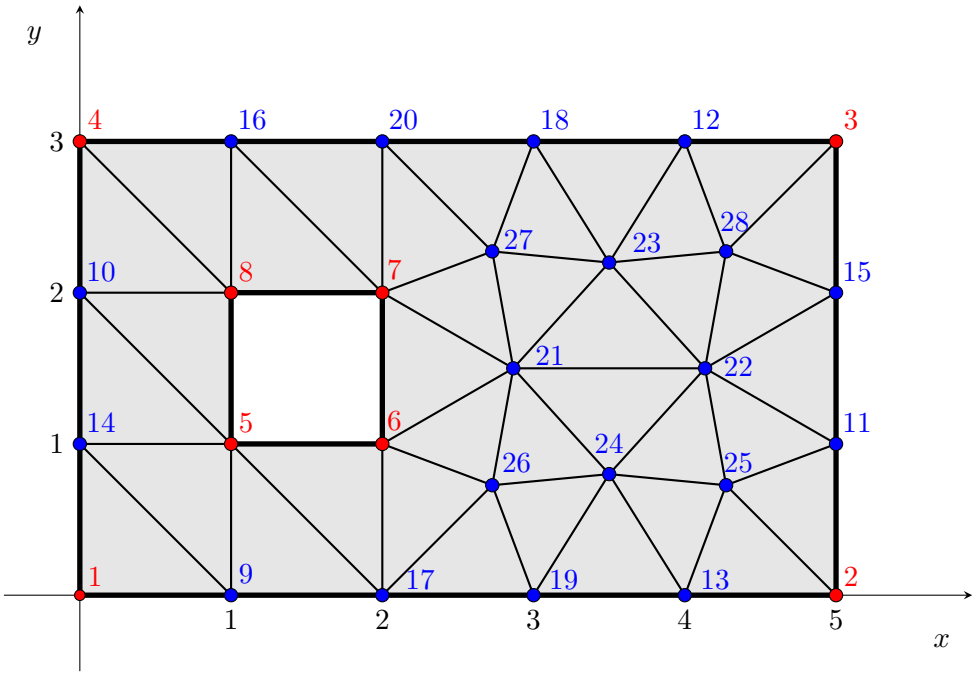
```
node = [                    % list of xy "node" coordinates
       0, 0   %1            % outer rectangle
       5, 0   %2
       5, 3   %3
       0, 3   %4
       1, 1   %5            % inner square
       2, 1   %6
       2, 2   %7
       1, 2   %8
          ] ;

edge = [                    % list of "edges" between nodes
```

```
        1, 2              % outer rectangle
        2, 3
        3, 4
        4, 1
        5, 6              % inner square
        6, 7
        7, 8
        8, 5 ] ;
%------------------------------------------ call mesh-gen
   h = 1;
  [p,e,t] = refine2(node,edge,[],[],h)
```

Kód 26: Vstupy pro volání funkce refine2

C.3 Výstupní hodnoty

Výstupními parametry funkce `[p,e,t] = refine2(node,edge,[],[],h)` je seznam všech uzlů `p`, viz obrázek 16. Seznam hran na hranici oblasti `e` a seznam všech trojúhelníků `t`.



Obrázek 16: Síť

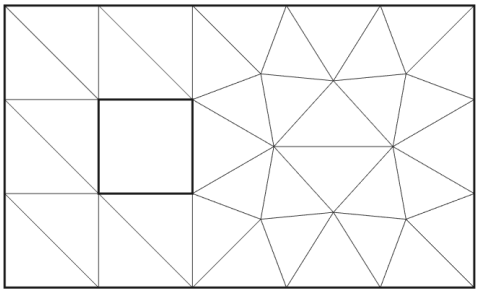
```
p =
         0         0  %1
    5.0000         0  %2
    5.0000    3.0000  %3
         0    3.0000  %4
    1.0000    1.0000  %5
     2.0000   1.0000  %6
    2.0000    2.0000  %7
    1.0000    2.0000  %8
    1.0000         0  %9
         0    2.0000  %10
    5.0000    1.0000  %11
    4.0000    3.0000  %12
    4.0000         0  %13
         0    1.0000  %14
       .
       .
       .
    5.0000    2.0000  %15
    1.0000    3.0000  %16
    2.0000         0  %17
    3.0000    3.0000  %18
    3.0000         0  %19
    2.0000    3.0000  %20
    2.8660    1.5000  %21
    4.1340    1.5000  %22
    3.5000    2.1994  %23
    3.5000    0.8006  %24
    4.2730    0.7270  %25
    2.7270    0.7270  %26
    2.7270    2.2730  %27
    4.2730    2.2730  %28
```

```
e =
     5     6
     8     5
     6     7
     7     8
     1     9
     4    10
     2    11
     3    12
     2    13
     1    14
     3    15
     4    16
    10    14
    11    15
     9    17
    12    18
    13    19
    16    20
    17    19
    18    20
```

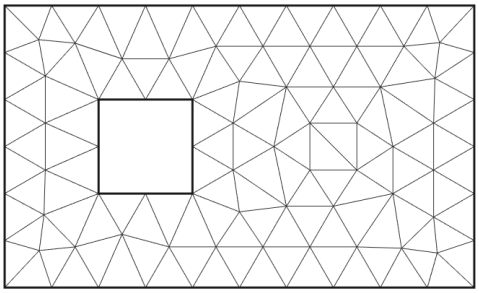
```
t =
     7     6    21
     8     7    16
    10     8     4
    25    24    13
     5     9    17
     6     5    17
    24    26    19
    26    17    19
    14     1     9
     6    26    21
     6    17    26
    24    21    26
    14     5    10
    14     9     5
     5     8    10
     4     8    16
    20     7    27
    27     7    21
     .
     .
     .
    20    16     7
    23    27    21
    18    20    27
    24    22    21
    18    27    23
    18    23    12
    23    28    12
    28    15     3
    22    23    21
    12    28     3
    15    28    22
    13    24    19
    22    28    23
    25    22    24
    11    15    22
     2    11    25
     2    25    13
    11    22    25
```

C.4 Srovnání sítí

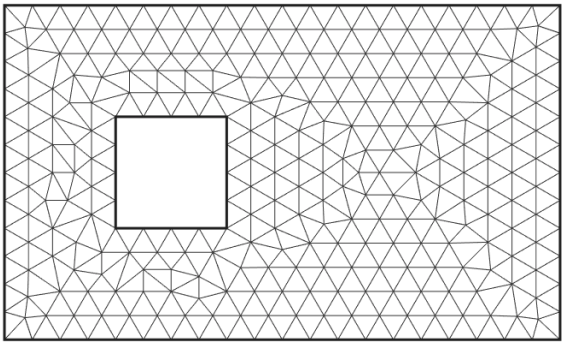
Na obrázcích 17–20 vidíme vygenerované sítě pro různé tvary a hodnoty `h`.



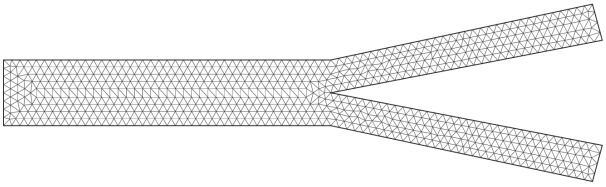
Obrázek 17: Síť pro `h=1`



Obrázek 18: Síť pro `h=0.5`



Obrázek 19: Síť pro `h=0.25`



Obrázek 20: Síť pro `h=0.5`

D ISO2MESH

D.1 Základní informace

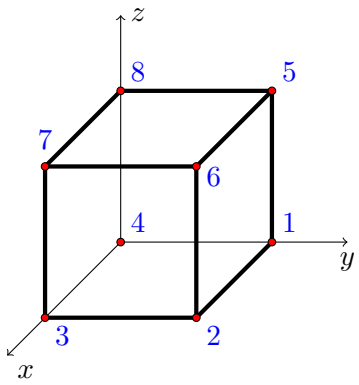
Balík ISO2MESH generuje sítě pro 3D objekty a je dostupný na adrese:
http://iso2mesh.sourceforge.net/cgi-bin/index.cgi

D.2 Zadání geometrie oblasti a generování sítě

Těleso s polygonální hranicí zadáme pomocí souřadnic jednotlivých vrcholů tělesa.

Například krychli na obrázku 21 popíšeme osmi vrcholy. Do vstupní proměnné `node` uvedeme x, y, z souřadnice těchto vrcholů. Plochy tvořící hranici tělesa uvedeme do proměnné `fc`, kde každou plochu popíšeme indexy příslušných vrcholů. Druhým parametrem máme možnost popsat jednotlivé plochy. V tomto příkladě budeme chtít (například kvůli zavádění různých okrajových podmínek) očíslovat plochy tak, že dolní podstava bude mít číslo 2, horní podstava 3 a všechny ostatní plochy 1.

V posledním řádku skriptu je uvedeno volání funkce `surf2mesh`, která vygeneruje síť. Její hustotu upřesníme pomocí paramatru `h`, kterým určíte maximální objem prvku.



Obrázek 21: Popis geometrie tělesa

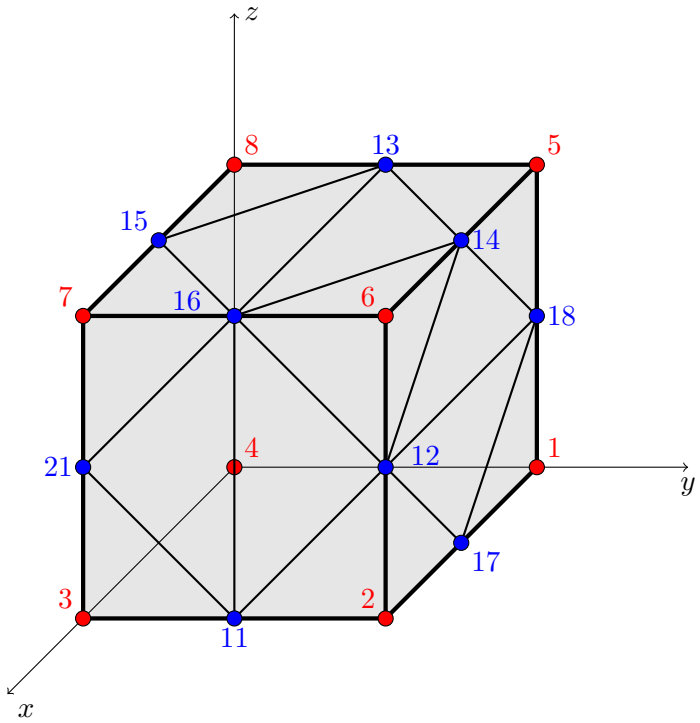
```
node =
    0     1     0;   %1
    1     1     0;   %2
    1     0     0;   %3
    0     0     0;   %4
    0     1     1;   %5
    1     1     1;   %6
    1     0     1;   %7
    0     0     1];  %8
```

```
% last number in fc is the area number
fc{1}={[4 8 5 1],1};
fc{2}={[2 6 7 3],1};
fc{3}={[1 5 6 2],1};
fc{4}={[3 7 8 4],1};
fc{5}={[1 2 3 4],2}; % lower base
fc{6}={[5 6 7 8],3}; % upper base
%------------------------------------------------ call mesh-gen
h = 0.5;
[p,t,e]=surf2mesh(node,fc,min(node),max(node),1,h,[0 0 1],[],0);
```

Kód 27: Vstupy pro volání funkce surf2mesh

D.3 Výstupní hodnoty

Výstupem příkazu `[p,e,t] = surf2mesh(node,fc,min(node),max(node),1,h,[0 0 1],[],0)` jsou souřadnice všech uzlů `p`, viz obrázek 22. Dále indexy vrcholů všech čtyřstěnů `t`. Všimněte si, že matice `t` má pět sloupců, poslední hodnota popisuje číslo části tělesa, pokud se skládá z více částí. Další výstupem `e` jsou indexy vrcholů trojúhleníků, ležících na hranici tělesa. V posledním sloupci je uvedeno číslo plochy, tak jak jsme je zadali v proměnné `fc`.



Obrázek 22: Síť

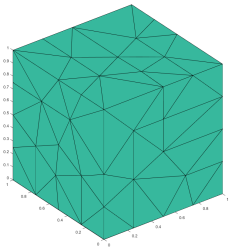
```
p =
         0    1.0000         0   %1
    1.0000    1.0000         0   %2
    1.0000         0         0   %3
         0         0         0   %4
         0    1.0000    1.0000   %5
    1.0000    1.0000    1.0000   %6
    1.0000         0    1.0000   %7
         0         0    1.0000   %8
         0         0    0.5002   %9
         0    0.5000         0   %10
    1.0000    0.5001         0   %11
    1.0000    1.0000    0.5005   %12
         0    0.5002    1.0000   %13
    0.5002    1.0000    1.0000   %14
    0.5005         0    1.0000   %15
    1.0000    0.5000    1.0000   %16
    0.4996    1.0000         0   %17
         0    1.0000    0.5004   %18
    0.5005         0         0   %19
    1.0000    0.5000    0.5000   %20
    1.0000         0    0.5002   %21
```

```
t =
    12     6    14    16     1
     4     9    19    10     1
    13     5    14    18     1
     8     9    13    15     1
    10    17     1    18     1
     3    20    19    21     1
    19     3    11    20     1
    14    12    16    20     1
    19     9    21    20     1
    11     2    17    20     1
    17     2    12    20     1
    17    12    14    20     1
    20    16    15    21     1
    16     7    15    21     1
    21     9    15    20     1
    16    13    14    20     1
    17    10    11    20     1
     9    19    10    20     1
    10    19    11    20     1
    16    15    13    20     1
    13    15     9    20     1
    13     9    10    20     1
    14    13    18    20     1
    18    13    10    20     1
    18    17    14    20     1
    10    17    18    20     1
```

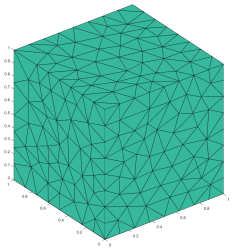
```
e =
     1    18    10     1
    13     9    10     1
    19    21     3     1
    18    17    14     1
    17     2    12     1
     7    21    15     1
     4     9    19     1
    17    10    11     2
    10    19    11     2
     6    16    14     3
    16    13    14     3
    19     9    21     1
     8     9    13     1
     4    10     9     1
    19    10     4     2
    11     2    17     2
     3    21    20     1
    12    16     6     1
     5    18    14     1
    13    18     5     1
    13     5    14     3
    16    15    13     3
    12     6    14     1
     8    15     9     1
    13    15     8     3
    16     7    15     3
    10    17     1     2
    17    12    14     1
    17    18     1     1
    18    13    10     1
    19     3    11     2
    21     9    15     1
     3    20    11     1
    12    20    16     1
    20    21    16     1
    16    21     7     1
    11    20     2     1
     2    20    12     1
```

D.4 Srovnání sítí

Na obrázcích 23–24 vidíme vygenerované sítě pro různé hodnoty `h`.



Obrázek 23: Síť pro h=0.01



Obrázek 24: Síť pro h=0.001

E Kódy pro úlohu (25),(23)-(24)

```
function [A,B,E,b,c,a]=assemblyP1bP1(p,t,alpha,nu,f)
np=size(p,1); nt=size(t,1);
A=sparse(2*np,2*np); B=sparse(np,2*np); E=sparse(np,np);
b=zeros(2*np,1); c=zeros(np,1); a=zeros(np,1);
Mel=(1/12)*[2 1 1; 1 2 1; 1 1 2]; % element mass matrix
for ih=1:nt
    itB=t(ih,1:3); it1=2*itB-1; it2=it1+1;
    x21=p(t(ih,2),1)-p(t(ih,1),1); y12=p(t(ih,1),2)-p(t(ih,2),2);
    x32=p(t(ih,3),1)-p(t(ih,2),1); y23=p(t(ih,2),2)-p(t(ih,3),2);
    x13=p(t(ih,1),1)-p(t(ih,3),1); y31=p(t(ih,3),2)-p(t(ih,1),2);
    tarea=(x21*y31-x13*y12)/2;
    xt=[y23; y31; y12]; yt=[x32; x13; x21]; mt=(3/20)*alpha*tarea*ones(3,1);
    omega=(81/40)*nu*(y23^2+x32^2-y31*y12-x13*x21)/tarea+(81/280)*alpha*tarea;
    % element matrices and vectors
    xtxtytyt=xt*xt'+yt*yt';
    Ah=alpha*tarea*Mel+nu*(0.25/tarea)*xtxtytyt-mt*mt'/omega;
    B1h=-(1/6)*[xt';xt';xt']-(9/40/omega*xt)*mt';
    B2h=-(1/6)*[yt';yt';yt']-(9/40/omega*yt)*mt';
    Eh=(81/1600/omega)*xtxtytyt;

    f1t=(f(it1(1))+f(it1(2))+f(it1(3)))/3;
    f2t=(f(it2(1))+f(it2(2))+f(it2(3)))/3;
    f1b=(9/20)*tarea*f1t; f2b=(9/20)*tarea*f2t;
    b1h=(1/3)*tarea*[f1t;f1t;f1t]-(f1b/omega)*mt;
    b2h=(1/3)*tarea*[f2t;f2t;f2t]-(f2b/omega)*mt;
    ch=-(9/40/omega)*(f1b*xt+f2b*yt);
    ah=(tarea/3)*[1;1;1];

    % assembly global data
    A(it1,it1)=A(it1,it1)+Ah; A(it2,it2)=A(it2,it2)+Ah;
    B(itB,it1)=B(itB,it1)+B1h; B(itB,it2)=B(itB,it2)+B2h;
    E(itB,itB)=E(itB,itB)+Eh;
    b(it1)=b(it1)+b1h; b(it2)=b(it2)+b2h;
    c(itB)=c(itB)+ch; a(itB)=a(itB)+ah;
end
```

Kód 28: Standardní sestasvení ve 2D

```
function [A,B,E,b,c,a]=assemblyP1bP1_vec(p,t,alpha,nu,f)

tt=2*t; tt=[tt-1;tt];
np=size(p,1); np2=2*np;

A=sparse(np2,np2); B=sparse(np,2*np); E=sparse(np,np);
b=zeros(np2,1); c=zeros(np,1); a=zeros(np,1);

x32=p(t(:,3),1)-p(t(:,2),1); y23=p(t(:,2),2)-p(t(:,3),2);
x13=p(t(:,1),1)-p(t(:,3),1); y31=p(t(:,3),2)-p(t(:,1),2);
x21=p(t(:,2),1)-p(t(:,1),1); y12=p(t(:,1),2)-p(t(:,2),2);
tarea=(x21.*y31-x13.*y12)/2; alta=alpha*tarea;
xt=[y23 y31 y12]; yt=[x32 x13 x21];
omega=(81/40)*nu*(y23.^2+x32.^2-y31.*y12-x13.*x21)./tarea+(81/280)*alta;
% Mass matrix
aux=alta/12;
for i=1:3
    for j=1:i
        A=A+sparse(tt(:,i),tt(:,j),[aux;aux],np2,np2);
        A=A+sparse(tt(:,j),tt(:,i),[aux;aux],np2,np2);
    end
end
% Diffusion matrix
n4ta=(nu/4)./tarea;
for i=1:3
    n4taxi=n4ta.*xt(:,i); n4tayi=n4ta.*yt(:,i);
    for j=i:3 % A11, A22=A11
        n4taxixj=n4taxi.*xt(:,j); n4tayiyj=n4tayi.*yt(:,j);
        aux=n4tayiyj+n4taxixj;
        if i==j
            A=A+sparse(tt(:,i),tt(:,j),[aux;aux],np2,np2);
        else
            A=A+sparse(tt(:,i),tt(:,j),[aux;aux],np2,np2);
            A=A+sparse(tt(:,j),tt(:,i),[aux;aux],np2,np2);
        end
    end
end
% Divergence matrix
```

```
for j=1:3
    aux=[xt(:,j)/6;yt(:,j)/6];
    for i=1:3
        B=B-sparse([t(:,i);t(:,i)],tt(:,j),aux,np,np2);
    end
end
% Bubble components elimination in A
mt=3/20*alta;
aux=mt.*mt./omega;
for i=1:3
    for j=1:3
        A=A-sparse(tt(:,i),tt(:,j),[aux;aux],np2,np2);
    end
end
% Bubble components elimination in B, E
aux=(9/40)./omega;
for i=1:3
    aux1=aux.*xt(:,i); aux1m=aux1.*mt; aux2=aux.*yt(:,i); aux2m=aux2.*mt;
    for j=1:3
        E=E+sparse(t(:,i),t(:,j),9/40*(aux1.*xt(:,j)+aux2.*yt(:,j)),np,np);
        B=B-sparse([t(:,i);t(:,i)],tt(:,j),[aux1z;aux2z],np,np2);
    end
end
% Right hand-side vector b
f1=f(1:2:np2); f2=f(2:2:np2);
f1=tarea.*(f1(t(:,1))+f1(t(:,2))+f1(t(:,3)))/9 ;
f2=tarea.*(f2(t(:,1))+f2(t(:,2))+f2(t(:,3)))/9;
for i=1:3
    b=b+sparse(tt(:,i),1,[f1;f2],np2,1);
end
% Bubble components elimination in b, c and a
aux=(27/20)./omega; f1b=aux.*f1; f2b=aux.*f2;
for i=1:3
    b=b-sparse(tt(:,i),1,[mt.*f1b;mt.*f2b],np2,1);
    c=c-sparse(t(:,i),1,(9/40)*(f1b.*xt(:,i)+f2b.*yt(:,i)),np,1);
    a=a+sparse(t(:,i),1,tarea/3,np,1);
end
```

Kód 29: Vektorizované sestavení ve 2D

```
function [A,B,E,b,c,a]=assemblyP1bP1_3(p,t,alpha,nu,f)
np=size(p,1); nt=size(t,1);
A=sparse(3*np,3*np); B=sparse(np,3*np); E=sparse(np,np);
b=zeros(3*np,1); c=zeros(np,1); a=zeros(np,1);
Mel=(1/20)*[2 1 1 1; 1 2 1 1; 1 1 2 1; 1 1 1 2]; % element mass matrix
for ih=1:nt
    itB=t(ih,1:4); it1=3*itB-2; it2=it1+1; it3=it1+2;
    x21=p(itB(2),1)-p(itB(1),1); y21=p(itB(2),2)-p(itB(1),2);
    x31=p(itB(3),1)-p(itB(1),1); y31=p(itB(3),2)-p(itB(1),2);
    x41=p(itB(4),1)-p(itB(1),1); y41=p(itB(4),2)-p(itB(1),2);
    x32=p(itB(3),1)-p(itB(2),1); y32=p(itB(3),2)-p(itB(2),2);
    x42=p(itB(4),1)-p(itB(2),1); y42=p(itB(4),2)-p(itB(2),2);
    z21=p(itB(2),3)-p(itB(1),3);
    z31=p(itB(3),3)-p(itB(1),3);
    z41=p(itB(4),3)-p(itB(1),3);
    z32=p(itB(3),3)-p(itB(2),3);
    z42=p(itB(4),3)-p(itB(2),3);
    xt=[z32*y42-y32*z42;y31*z41-z31*y41;z21*y41-y21*z41;y21*z31-z21*y31];
    yt=[x32*z42-z32*x42;z31*x41-x31*z41;x21*z41-z21*x41;z21*x31-x21*z31];
    zt=[y32*x42-x32*y42;x31*y41-y31*x41;y21*x41-x21*y41;x21*y31-y21*x31];
    tvolume=(x21*xt(2)+x31*xt(3)+x41*xt(4))/6;
    msgn=-sign(tvolume); tvolume=abs(tvolume); altv=alpha*tvolume;
    ah=0.25*[tvolume;tvolume;tvolume;tvolume];
    mt=(alpha*32/105)*ah;
    xtt=xt(2)*(xt(3)+xt(4))+xt(3)*xt(4)+yt(2)*(yt(3)+yt(4))+yt(3)*yt(4)...
        +zt(2)*(zt(3)+zt(4))+zt(3)*zt(4);
    omega=nu*2048/8505/tvolume*(xt(1)^2+yt(1)^2+zt(1)^2-xtt)+621/3940*altv;
    % element matrices and vectors
    xyzt=xt*xt'+yt*yt'+zt*zt';
    Ah=altv*Mel+(nu/36/tvolume)*xyzt-(1/omega*mt)*mt';
    B1h=msgn*(1/24*[xt';xt';xt';xt']+(16/315/omega*xt)*mt');
    B2h=msgn*(1/24*[yt';yt';yt';yt']+(16/315/omega*yt)*mt');
    B3h=msgn*(1/24*[zt';zt';zt';zt']+(16/315/omega*zt)*mt');
    Eh=(256/99225/omega)*xyzt;
    f1t=(f(it1(1))+f(it1(2))+f(it1(3))+f(it1(4)))/4;
    f2t=(f(it2(1))+f(it2(2))+f(it2(3))+f(it2(4)))/4;
    f3t=(f(it3(1))+f(it3(2))+f(it3(3))+f(it3(4)))/4;
    f1b=(96/315)*tvolume*f1t; f2b=(96/315)*tvolume*f2t;
```

```
    f3b=(96/315)*tvolume*f3t;
    b1h=f1t*ah-(f1b/omega)*mt; b2h=f2t*ah-(f2b/omega)*mt;
    b3h=f3t*ah-(f3b/omega)*mt;
    ch=msgn*(16/315)/omega*(f1b*xt+f2b*yt+f3b*zt);
    % assembly global data
    A(it1,it1)=A(it1,it1)+Ah; A(it2,it2)=A(it2,it2)+Ah;
    A(it3,it3)=A(it3,it3)+Ah;
    B(itB,it1)=B(itB,it1)+B1h; B(itB,it2)=B(itB,it2)+B2h;
    B(itB,it3)=B(itB,it3)+B3h;
    E(itB,itB)=E(itB,itB)+Eh;
    b(it1)=b(it1)+b1h; b(it2)=b(it2)+b2h; b(it3)=b(it3)+b3h;
    c(itB)=c(itB)+ch; a(itB)=a(itB)+ah;
end
```

Kód 30: Standardní sestavení ve 3D

```
function [A,B,E,b,c,a]=assemblyP1bP1(p,t,alpha,nu,f)
tt=3*t; tt=[tt-2;tt-1;tt]; np=size(p,1); np3=3*np;
f1=f(1:3:np3-2); f2=f(2:3:np3-1); f3=f(3:3:np3);
A=sparse(np3,np3); B=sparse(np,np3); E=sparse(np,np);
b=zeros(np3,1); c=zeros(np,1); a=zeros(np,1);
x21=p(t(:,2),1)-p(t(:,1),1); y21=p(t(:,2),2)-p(t(:,1),2);
x31=p(t(:,3),1)-p(t(:,1),1); y31=p(t(:,3),2)-p(t(:,1),2);
x41=p(t(:,4),1)-p(t(:,1),1); y41=p(t(:,4),2)-p(t(:,1),2);
x32=p(t(:,3),1)-p(t(:,2),1); y32=p(t(:,3),2)-p(t(:,2),2);
x42=p(t(:,4),1)-p(t(:,2),1); y42=p(t(:,4),2)-p(t(:,2),2);
z21=p(t(:,2),3)-p(t(:,1),3);
z31=p(t(:,3),3)-p(t(:,1),3);
z41=p(t(:,4),3)-p(t(:,1),3);
z32=p(t(:,3),3)-p(t(:,2),3);
z42=p(t(:,4),3)-p(t(:,2),3);
xt=[z32.*y42-y32.*z42,y31.*z41-z31.*y41,z21.*y41-y21.*z41,y21.*z31-z21.*y31];
yt=[x32.*z42-z32.*x42,z31.*x41-x31.*z41,x21.*z41-z21.*x41,z21.*x31-x21.*z31];
zt=[y32.*x42-x32.*y42,x31.*y41-y31.*x41,y21.*x41-x21.*y41,x21.*y31-y21.*x31];
tvolume=(x21.*xt(:,2)+x31.*xt(:,3)+x41.*xt(:,4))/6;
msgn=-sign(tvolume); tvolume=abs(tvolume); altv=alpha*tvolume; mt=8/105*altv;
xtt=xt(:,2).*(xt(:,3)+xt(:,4))+xt(:,3).*xt(:,4)+yt(:,2).*(yt(:,3)+yt(:,4))...
    +yt(:,3).*yt(:,4)+zt(:,2).*(zt(:,3)+zt(:,4))+zt(:,3).*zt(:,4);
```

```
omega=2048/8505*nu./tvolume.*(xt(:,1).^2+yt(:,1).^2+zt(:,1).^2-xtt)...
                                                        +621/3940*altv;
f1t=(f1(t(:,1))+f1(t(:,2))+f1(t(:,3))+f1(t(:,4)))/4;
f2t=(f2(t(:,1))+f2(t(:,2))+f2(t(:,3))+f2(t(:,4)))/4;
f3t=(f3(t(:,1))+f3(t(:,2))+f3(t(:,3))+f3(t(:,4)))/4;
f3b=96/315*tvolume;
f1b=f3b.*f1t; f2b=f3b.*f2t; f3b=f3b.*f3t;
auxM=altv/20; auxM=[auxM;auxM;auxM];
mtom=mt./omega;
auxR1=1/36*nu./tvolume; auxR2=mt.*mtom;
auxB=16/315*msgn.*mtom; auxB24=msgn/24;
auxE=256/99225./omega; ah=tvolume/4;
auxb=[f1t.*ah-f1b.*mtom;f2t.*ah-f2b.*mtom;f3t.*ah-f3b.*mtom];
auxc=16/315*msgn./omega;
for j=1:4
    auxB1j=auxB24.*xt(:,j); auxB2j=auxB24.*yt(:,j); auxB3j=auxB24.*zt(:,j);
    for i=1:4
      B=B+sparse([t(:,i);t(:,i);t(:,i)],tt(:,j),[auxB1j;auxB2j;auxB3j],np,np3);
    end
end
for i=1:4
    for j=1:i
      A=A+sparse(tt(:,i),tt(:,j),auxM,np3,np3)...
         +sparse(tt(:,j),tt(:,i),auxM,np3,np3);
    end
    auxB1i=auxB.*xt(:,i); auxB2i=auxB.*yt(:,i); auxB3i=auxB.*zt(:,i);
    for j=1:4
      xyzt=xt(:,i).*xt(:,j)+yt(:,i).*yt(:,j)+zt(:,i).*zt(:,j);
      auxR=auxR1.*xyzt-auxR2;
      A=A+sparse(tt(:,i),tt(:,j),[auxR;auxR;auxR],np3,np3);
      B=B+sparse([t(:,i);t(:,i);t(:,i)],tt(:,j),[auxB1i;auxB2i;auxB3i],np,np3);
      E=E+sparse(t(:,i),t(:,j),auxE.*xyzt,np,np);
    end
    a=a+sparse(t(:,i),1,ah,np,1);
    b=b+sparse(tt(:,i),1,auxb,np3,1);
    c=c+sparse(t(:,i),1,auxc.*(f1b.*xt(:,i)+f2b.*yt(:,i)+f3b.*zt(:,i)),np,1);
end
```

Kód 31: Vektorizované sestavení ve 3D